Universidad Autónoma Metropolitana
Unidad Azcapotzalco
División de Ciencias Básicas e Ingeniería
Licenciatura en Ingeniería en Computación

Sistema web para la búsqueda y ordenamiento de videojuegos a partir de la Web basado en el precio

Modalidad: Proyecto Tecnológico

Trimestre 2018 Otoño

Alejandro López López
206358631
lopez.alejandro21@gmail.com

José Alejandro Reyes Ortiz
Doctor en Ciencias de la Computación
Profesor Asociado
Departamento de Sistemas
jaro@correo.azc.uam.mx

Declaratoria

Yo, José Alejandro Reyes Ortiz, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca digital, así como en el Repositorio Institucional de la Universidad Autónoma Metropolitana Unidad Azcapotzalco.

José Alejandro Reyes Ortiz

Yo, Alejandro López López, doy mi autorización a la Coordinación de Servicios de Información de la Universidad Autónoma Metropolitana, Unidad Azcapotzalco, para publicar el presente documento en la Biblioteca Digital, así como en el Repositorio Institucional de la Universidad Autónoma Metropolitana, Unidad Azcapotzalco.

Alejandro López López

Resumen

La visualización consiste en transformar información en imágenes que facilitan la extracción de significado. A veces, esa información es cuantitativa, y la visualización creada puede ser llamada “de datos”, en ese caso, el objetivo del gráfico es permitir identificar patrones y tendencias que serían invisibles si esos mismos datos se presentan en una tabla de datos plasmada en una página web.

Este proyecto utiliza la idea anterior al crear una página web que, partiendo de una base de datos, facilita el observar la relación entre el precio de los videojuegos en diferentes páginas web y los resultados de todos los videojuegos encontrados serán visualizados a través del uso de una tabla.

Tabla de contenido

Índice de Figuras

Introducción

En la actualidad los videojuegos han tenido un gran auge en la población tanto infantil como adulta, estos han evolucionado de tal manera que desde la década de los 70's en lo que se realizaban videojuegos de 8 bits para consolas como Atari que fue una de las primeras consolas en lanzar a la venta juegos de 8 bits, la compañía Mattel intento lanzar consolas pero no duro mucho en el mercado, siendo de las primeras en cerrar su proyecto de videojuegos.
Hoy en día los videojuegos han avanzado tanto grafica como tecnológicamente llegando a un punto en el que se pueda jugar en realidad virtual, simulando una interacción con la inteligencia artificial programada dentro del videojuego.

Por todo lo ya mencionado, este proyecto tiene como objetivo la implementación de un sistema web que tendrá parte de este proyecto presentado para brindar apoyo a la comunidad de video jugadores mediante la clasificación de los videojuegos primeramente se clasificaran los videojuegos de menor precio incluyendo promociones y descuentos. Este proyecto iniciara a partir de los datos generados de distintas páginas web con la finalidad de procesar dichos datos y donde es común ver la venta de videojuegos ya que es por estos medios donde los video jugadores adquiere con mayor frecuencia estos productos ya sea porque están más baratos que en establecimientos o por que los pueden adquirir lo más cercano a su domicilio.

Antecedentes

Proyectos de integración Internos

1. Sistema web para la geolocalización de incidentes delictivos a partir de publicaciones de twitter[1]

En este proyecto toda la implementación es para extraer información de la página de twitter y clasificar la información por tipo de delitos que sucede tanto en la Ciudad de México como en el Estado de México

2. Sistema web para gestionar incidentes delictivos [2]

En esta aplicación los usuarios registrados pueden reportar delitos que pueden ser tales como: robo, homicidio, suicidio, explotación sexual, violación, recopila la información y la clasifica, la cual es publica para que los usuarios registrados tengan acceso para consultar dicha información.

3. Sistema concurrente para la extracción sobre suicidios en México a partir de tweets[3]

La propuesta tiene como objetivo crear una aplicación concurrente, basado en hilos los cuales ayudaran a la extracción de tweets que contengan en su texto los tipos de delitos generando una extracción de datos y clasificándolos para su consulta posteriormente por los usuarios.

Propuestas de Integración Externos

Software

4. Trivago. [4]

En este sitio web los usuarios pueden realizar una búsqueda de viajes en la cual muestra el mejor precio de vuelos y hoteles publicados en diferentes sitios web.

5. Kayak. [5]

Sitio web que realiza búsquedas para realizar viajes encontrar precios económicos de vuelos, hoteles y/o alquiler de automóviles.

6. Gocdkeys. [6]

En este sitio web los usuarios con ubicación en toda Europa pueden realizar una búsqueda de precios de videojuegos y clasificarla a partir del más económico.

Justificación

En la actualidad los jóvenes y adultos están inmersos en el mundo de los videojuegos y tener un sitio donde puedan encontrar información de los precios ya sea bajos o en oferta de los videojuegos puede ser una herramienta útil, ya que esta podrá llegar a ahorrar tiempo en la búsqueda por cada página de compra en línea que se tenga que visitar para hallar el videojuego deseado a bajo costo.

Cabe mencionar que los datos fueron extraídos de 3 páginas que tienen mayor venta de videojuegos estos se encuentran en las plataformas de Amazon, EBay y MercadoLibre y serán almacenados para su uso, estos datos se representan de manera general priorizando el bajo costo del videojuego o si este se encuentra en una oferta especial, por ello es necesario tener información del juego deseado para una búsqueda exitosa, con la finalidad de tener una extracción de datos de los videojuegos tales como es su precio/consolas/si es usado o nuevo/pagina donde se vende el videojuego.

Esta propuesta tiene la finalidad de recolectar datos de diferentes sitios webs, almacenarlos en una base de datos y mostrarlos al usuario de una manera clara y concisa, esto nos lleva a que se ahorre tiempo en estar visitando cada página web para encontrar la mejor opción a la hora de comprar un videojuego.

Objetivos

Objetivo General

Diseñar e implementar un sistema web para la búsqueda y extracción de información sobre videojuegos a partir de tres sitios web para generar una vista ordenada basada en el precio.

Objetivos Específicos

1. Diseñar e implementar un módulo para la búsqueda y extracción de datos de videojuegos a partir de tres páginas web usando patrones numéricos y de texto.
2. Diseñar e implementar un módulo para el almacenamiento de la información extraída (datos del videojuego, datos del desarrollador, precio, condición y referencia del sitio).
3. Implementar un módulo para generar una vista ordenada, basada en el precio, con los datos de los videojuegos.

Marco Teórico

Visualización

No hace mucho tiempo el término visualización significaba construir una imagen visual en la mente (Shorter Oxford English Dictionary, 1972). Ahora, el término ha cambiado a algo más parecido a: la representación gráfica de datos o conceptos. En consecuencia, de ser una construcción interna de la mente, la visualización se ha convertido en un artefacto externo que apoya la toma de decisiones.

Las visualizaciones tienen un rol pequeño pero crucial y creciente en los sistemas cognitivos. Las representaciones visuales proveen el ancho de banda más amplio entre el canal formado por la computadora y los humanos. Adquirimos más información a través de la visión que utilizando los otros sentidos combinados. Los 20 billones o más de neuronas del cerebro dedicadas a analizar la información visual proveen un mecanismo de búsqueda-de-patrones que es un componente fundamental en la mayoría de nuestra actividad cognitiva. Mejorar los sistemas cognitivos a menudo significa amarrar el lazo entre una persona, las herramientas basadas-en-computadora, y otras personas.

Uno de los grandes beneficios de la visualización de datos es la enorme cantidad de información que puede ser interpretada si es presentada correctamente.

Servlets

Los Servlets son módulos escritos en Java que se utilizan en un servidor, que puede ser o no ser servidor web, para extender sus capacidades de respuesta a los clientes al utilizar las potencialidades de Java. Los Servlets son para los servidores lo que los applets para los navegadores, aunque los servlets no tienen una interfaz gráfica.
Los servelts pueden ser incluidos en servidores que soporten la API de Servlet (ver servidores). La API no realiza suposiciones sobre el entorno que se utiliza, como tipo de servidor o plataforma, ni del protocolo a utilizar, aunque existe una API especial para HTTP.

Primero es necesario señalar que el servlet será del tipo HTTP por lo que se extiende de la clase **HttpServlet**. Al extender de esta clase es necesario definir el método **doGet** para responder la petición. Este método recibe los parámetros dados por el cliente a través de la clase **HttpServletRequest** y encapsula la respuesta que se le dará al cliente a través de la clase **HttpServletResponse**. El servlet puede retornar al cliente cualquier tipo de información, desde texto plano hasta un ejecutable, por lo que es necesario señalar inicialmente qué tipo de respuesta se dará a través del método **setContentType**. Luego se obtiene el objeto para poder escribir texto al cliente a través del método **getWriter** con el cual se puede retornar una página web llamado sucesivamente el método **println** hasta terminar con **close**.

Java WEB CRAWLER

Es un programa informático que inspecciona las paginas web. Una de sus funciones principales es la de crear una copia de las paginas web requeridas para ser procesadas posteriormente por un motor de búsqueda que indexa las paginas proporcionando un sistema de búsqueda más rápido.

El web crawler visita la página de una lista de URL's identifica hiperenlaces de dichas páginas.

Tiene diferentes tareas como son:

Crear el índice de un motor de búsqueda.
Analizar los enlaces para buscar links rotos.
Recolectar información especifica como pueden ser precios de productos.

JDOM

JDOM es una API desarrollada específicamente para Java y da soporte al tratamiento de XML: parseo, búsquedas, modificación, generación y serialización. Es un modelo similar a DOM, pero no está creado ni modelado sobre DOM. Se trata de un modelo alternativo. La principal diferencia es que DOM fue desarrollado para que fuera independiente del lenguaje, mientras que JDOM está creado y optimizado específicamente para Java. Esto imprime a JDOM las ventajas inherentes a Java, lo que lo hacen que sea una API más eficiente y más natural de usar para el desarrollador Java y por tanto requiere un menor coste de aprendizaje.

Al igual que DOM, JDOM genera un árbol de nodos al parsear un documento XML. En cuanto a lo tipos de nodos, son similares a los de DOM, aunque algunos cambia el nombre ligeramente. La jerarquía de clases también se ve algo modificada.

Mientras que en DOM todo hereda de la clase Node, en JDOM casi todo hereda de la clase Content, que se encuentran en el paquete org.jdom. Digo casi todo porque ni Document ni Attribute (Attr en DOM) heredan de Content. La razón es que Document no se considera un contenido, sino más bien el continente y Attribute tampoco se considera un nodo, sino una propiedad de los nodos tipo Element.

Otra diferencia entre DOM y JDOM es que mientras en DOM todos estos tipos de nodos eran interfaces, en JDOM son clases y por tanto para crear un objeto de cualquier tipo basta con usar el operador new.

Desarrollo del Proyecto

Desarrollo de la base de datos

Este proyecto se basa en datos de videojuegos que son extraídos es diferentes páginas web, los datos proporcionados por las paginas son almacenados en una base de datos y son los siguientes:

Nombre del Videojuego
Clasificación
Precio
Plataforma
Dirección de la pagina

La base de datos a usar se muestra en la Figura 1.

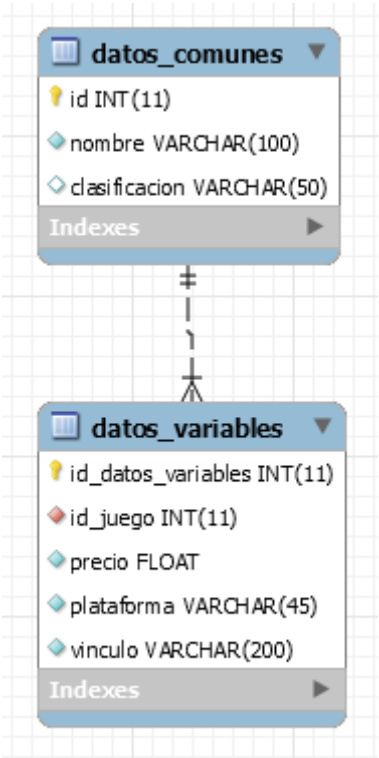

Figura 1. Modelo de la Base de Datos

Diseño del Proyecto

La funcionalidad del Proyecto se muestra en la Figura 2.

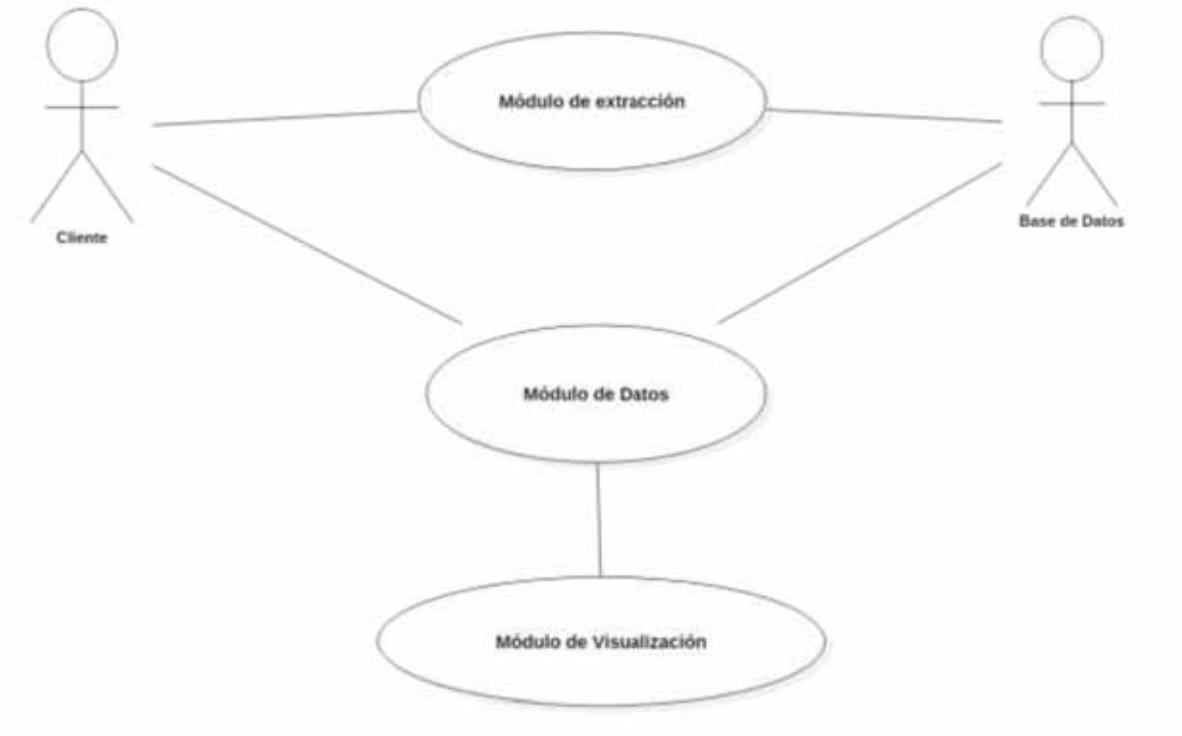

Figura 2. Funcionalidades del Proyecto

Módulo de extracción:

Este módulo se encarga de extraer toda la información de los videojuegos al momento de realizar la búsqueda.

Módulo de datos:

Realizará el almacenamiento de todos los datos recopilados de los distintos sitios web.

Módulo de visualización:

El objetivo de este módulo es el de mostrar al cliente una vista ordenada en un tablero de todos los datos buscados en los sitios web y que se encuentran almacenados en la base de datos.

Especificación Técnica

El proyecto se realizará mediante el lenguaje Java con entorno de desarrollo Netbeans 8.3 o posterior.

Para el ordenamiento de los videojuegos en los subtipos planteados en la Introducción, se llevará a cabo el diseño e implementación de una base de datos para determinar con patrones de precio que en los sitios web se hace mención. En el análisis y ordenamiento se hará uso de la interfaz JDBC (*Java DataBase Connectivity*) para tener acceso a la Base de Datos implementada en MySQL 5.5 o posterior.

Para la extracción de datos se utilizará un extractor de información (*Web Crawler)*, este será programado en java y su función será la de extraer todos los datos de los videojuegos que se encuentran en los sitios web ya mencionados.

Una vez que se tengan los datos almacenados se procederá a un ordenamiento en la base de datos de acuerdo a su precio del videojuego, se usará un *Web Crawler* en java para la extracción de datos en diferentes páginas web.

El proyecto se dará por concluido una vez que:

- El módulo de extracción recopile toda la información de los videojuegos al momento de realizar la búsqueda.
- El módulo de datos realice el almacenamiento de los datos recopilados de los sitios web.
- El módulo de presentación muestre una vista ordenada en un tablero de todos los datos buscados en los sitios web.

Página web (motor de búsqueda)

Esta página se compone de 2 ventanas en la cual la primera página muestra el motor de búsqueda en donde el cliente proporciona el nombre del videojuego:

Resultados (Figura 3 y 4)

Figura 3. Página principal motor de búsqueda

En esta parte de la página principal es donde el cliente escribe el nombre del videojuego y empieza a trabajar el java crawler junto con el Jdom, el primero saca información en XML la almacena en un archivo y el Jdom se encarga de buscar en dicho archivo los datos que requiere el cliente que son el precio, plataforma, y el link de la página donde está a la venta después entra la base de datos donde se encuentra almacenada la información de los videojuegos que se requiera por parte del cliente.

Figura 4. Resultados de búsqueda

En la figura anterior podemos observar el listado de resultados ordenados de menor a mayor precio y el link de donde se puede obtener para su compra.

Conclusiones

La utilidad de la visualización de datos se puede apreciar de buena manera en este proyecto, al permitir observar de manera ordenada a través del uso de un tablero y partiendo de los datos almacenados en la base de datos la información de los videojuegos buscados por el cliente.

Este proyecto desarrollado constituye una excelente forma de ahorrar tiempo al buscar un videojuego y su precio en distintas paginas web ya que las engloba en una sola y se puede observar que el ordenamiento de los datos es por el precio del videojuego económico al más costoso.

Y también permite visualizar el link de la página en donde se encuentra a la venta el videojuego para así facilitar el ir al sitio.

Referencias

[1] Amazon, S.L. búsqueda de videojuegos e información de sus precios para su ordenamiento http://www.amazon.com.mx/
[2] MercadoLibre página donde se buscaran datos para ser comparados con otras páginas web, su página web esta Disponible en:
http://www.mercadolibre.com.mx/ofertas
[3] EBay es otra de las páginas donde se recopilará la información de los videojuegos para su comparación en cuanto a precios, su página web está disponible en:
http://ebay.com
[4] A. López Arteaga, "Sistema web para la geolocalización de incidentes delictivos a partir de publicaciones en Twitter", Proyecto terminal, División de Ciencias Básicas e Ingeniería, Universidad Autónoma Metropolitana Azcapotzalco, México, 2017.
[5] R. A. Ruvalcaba Flores, "Sistema Web para gestionar incidentes delictivos", Proyecto terminal, División de Ciencias Básicas e Ingeniería, Universidad Autónoma Metropolitana Azcapotzalco, México, 2017.
[6] Ma.G. Rodríguez Ramos, "Sistema concurrente para la extracción de información sobre suicidios en México a partir de Twitter", Propuesta de Proyecto terminal, División de Ciencias Básicas e Ingeniería, Universidad Autónoma Metropolitana Azcapotzalco, México, 2017.
[7] Developer&Co., "Alerta Ciudadana", [Online] Disponible en: https://play.google.com/store/apps/details?id=com.alertaCiudadana.www&hl=es
[8] Trivago página que extrae información de otras páginas de hoteles agencias de viaje para mostrar los mejores precios su página está disponible en:
http://www.trivago.com.mx/CompararPrecios
[9] Kayak esta página busca en cientos de páginas web de viajes y da información acerca de los precios más baratos, su página web está disponible en:
http://www.kayak.com.mx

Anexo A: Código Fuente Documentado

Html en cascada(CCS)

Descripción:

Con este código realizamos el diseño de la página de inicio con el cuadro de texto para realizar la búsqueda del videojuego

```
<body>
        <center>
                <!-- Titulo -->
                <h1 Style="color: red">BUSCA EL PRECIO DE TU VIDEOJUEGO</h1>
                <!-- Campo para el nombre del videojuego -->
                <!-- <form name="Busqueda" method="post"> -->
                <input type="text" size="50" name="videojuego" autocomplete="off"
                        placeholder="Introduce el nombre de tu videojuego">
                <!-- Boton para enviar la consulta -->
                <input type="submit" name="buscar">
                <!-- </form> -->
        </center>
</body>
```

Clase para generar vista de tabla

Descripción:

Con esta clase podremos realizar la vista de lo que contenga la tabla en MySQL y asi poderla plasmar en la pagina web, esta estará escrita en la clase doPost del Servlet

```
protected void doPost(HttpServletRequest request, HttpServletResponse response)
                        throws ServletException, IOException {

class Videojuegos implements TableVideojuegos
{
   public void setValueAt (Object dato, int fila, int columna) {
       // Obtenemos los videojuegos de la fila indicada
      videojuego aux = (Videojuego)datos.get (fila);
      switch (columna) {
        // Nos pasan el nombre.
        case 0:
           aux.nombre = (String)dato;
           break;
        // Nos pasan el clasificacion.
        case 1:
           aux.clasificacion = (String)dato;
           break;
```

```
        // Nos pasan el precio.
        case 2:
            aux.precio = ((Integer)dato).intValue();
            break;
        // Nos pasan la plataforma.
         case3:
         aux.plataforma = (String)dato;
         break;
        // Nos pasan el vinculo.
         case4:
         aux.vinculo = (String)dato
    }
}
```

Conector a la base de datos

Descripción:

Con el conector a la base de datos desde java podremos acceder a la base de datos para poder realizar la vista de la tabla, el guardado de datos desde el Java Crawler en conjunto con el Jdom.

```
private static final long serialVersionUID = 1L;
public static final String URL = "jdbc:mysql://localhost:3306/datos_videojuegos";
public static final String USERNAME = "root";
public static final String PASSWORD = "sakuranbo";
public static void main(String[] args) {

        try {

                        // 1.- Conexion
Connection miBase = (Connection) DriverManager.getConnection(URL, USERNAME,
PASSWORD);

                        // 2.- Objeto statement

Statement miStatement = (Statement) miBase.createStatement();

                        // 3.- Ejecutar SQL

ResultSet miResultado = miStatement.executeQuery("SELECT * FROM
DATOS_COMUNES");

                        // 4.- Recorrer el ResultSet

while (miResultado.next()) {
```

```
System.out.println(miResultado.getString("NOMBRE") + " " +
miResultado.getString("CLASIFICACION"));
                        }

                } catch (Exception e) {
                        System.out.println("No conecta!!");

                        e.printStackTrace();
                }
```

Búsqueda del videojuego con el web crawler

Descripción:

Con esta clase podremos realizar la búsqueda del videojuego escribiendo su nombre, este nos buscara en las paginas de internet en este caso (Amazon) el videojuego que le hayamos escrito en la página de inicio.

```
public class Busqueda {

   String amazon(String busqueda) {

      int i=0;
      String plus="";

      System.out.println("busqueda");

      busqueda=busqueda.toLowerCase();
      String [] palabras = busqueda.split(" ");
      String
url="https://www.amazon.com.mx/s/ref=nb_sb_ss_i_3_13?__mk_es_MX=%C3%85M%C
3%85%C5%BD%C3%95%C3%91&url=search-alias%3Daps&field-keywords=";

      if (palabras.length>0) {
         while (i<palabras.length) {
            System.out.println(palabras[i]);
            plus=plus+palabras[i];
            i++;
         }
      }
      return url+plus;
   }

}
```

Clase del crawler para buscar las url's

Descripción:

Con esta clase podremos buscar todos los links que se encuentran en la página señalada (Amazon) nos dará el total de links encontrados con el videojuego.

```
public class Crawler {
   private static final int MAX_PAGES_TO_SEARCH = 50;
   private Set<String> pagesVisited = new HashSet<String>();
   private List<String> pagesToVisit = new LinkedList<String>();
   public void search(String url, String searchWord)
   {
    while(this.pagesVisited.size() < MAX_PAGES_TO_SEARCH)
    {
       String currentUrl;
       PataDeCrawler leg = new PataDeCrawler();
       if(this.pagesToVisit.isEmpty())
       {
          currentUrl = url;
          this.pagesVisited.add(url);
       }
       else
       {
          currentUrl = this.nextUrl();
       }
       leg.crawl(currentUrl); // Lots of stuff happening here. Look at the crawl method in
                         // SpiderLeg
       boolean success = leg.searchForWord(searchWord);
       if(success)
       {
          System.out.println(String.format("**Success** Word %s found at %s",
searchWord, currentUrl));
          break;
       }
       this.pagesToVisit.addAll(leg.getLinks());
    }
    System.out.println("\n**Done** Visited " + this.pagesVisited.size() + " web page(s)");
 }
private String nextUrl()
 {
    String nextUrl;
    do
    {
       nextUrl = this.pagesToVisit.remove(0);
    } while(this.pagesVisited.contains(nextUrl));
    this.pagesVisited.add(nextUrl);
```

```
    return nextUrl;
  }

}
```

Clase para probar el Crawler en cuanto a búsqueda de videojuego y los links que contienen el videojuego deseado

Descripción:

Aquí escribiremos el nombre del videojuego y este ejecutara la búsqueda

```
public class CrawlerTest
{
  public static void main(String[] args)
   {
      Crawler spider = new Crawler();
      Busqueda busqueda = new Busqueda();
      spider.search(busqueda.amazon("Spyro"), "Spyro");
   }
}
```

Con la Clase Pata de Crawler obtendremos todos los links de la búsqueda realizada con el test

```
public class PataDeCrawler
{

   private static final String USER_AGENT =
         "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.1 (KHTML, like
Gecko) Chrome/13.0.782.112 Safari/535.1";
   private List<String> links = new LinkedList<String>();
   private Document htmlDocument;

   public boolean crawl(String url)
   {
      try
      {
         Connection connection = Jsoup.connect(url).userAgent(USER_AGENT);
         Document htmlDocument = connection.get();
         this.htmlDocument = htmlDocument;
         if(connection.response().statusCode() == 200) // 200 is the HTTP OK status code
                                        // indicating that everything is great.
         {
            System.out.println("\n**Visiting** Received web page at " + url);
         }
         if(!connection.response().contentType().contains("text/html"))
```

```
            {
                System.out.println("**Failure** Retrieved something other than HTML");
                return false;
            }
            Elements linksOnPage = htmlDocument.select("a[href]");
            System.out.println("Found (" + linksOnPage.size() + ") links");
            for(Element link : linksOnPage)
            {
                this.links.add(link.absUrl("href"));
            }
            return true;
        }
        catch(IOException ioe)
        {

            return false;
        }
    }

    public boolean searchForWord(String searchWord)
    {

        if(this.htmlDocument == null)
        {
            System.out.println("ERROR! Call crawl() before performing analysis on the
document");
            return false;
        }
        System.out.println("Searching for the word " + searchWord + "...");
        String bodyText = this.htmlDocument.body().text();
        return bodyText.toLowerCase().contains(searchWord.toLowerCase());
    }

    public List<String> getLinks()
    {
        return this.links;
    }

}
```

JDom parser

Descripción:

Con el Jdom parser podremos extraer la información requerida filtrándola desde un archivo XML que nos brinda el Java Crawler

```
public class JDomParserDemo {

  public static void main(String[] args) {

    try {
      File inputFile = new File("input.txt");
      SAXBuilder saxBuilder = new SAXBuilder();
      Document document = saxBuilder.build(inputFile);
      System.out.println("Root element :" + document.getRootElement().getName());
      Element classElement = document.getRootElement();

      List<Element> studentList = classElement.getChildren();
      System.out.println("----------------------------");

      for (int temp = 0; temp < studentList.size(); temp++) {
        Element game = gameList.get(temp);
        System.out.println("\nCurrent Element :"
          + game.getName());
        Attribute attribute =  game.getAttribute("rollno");
        System.out.println("Game roll no : "
          + attribute.getValue() );
        System.out.println("Game Name: "
          + student.getChild("gamename").getText());
        System.out.println("Plataforma: "
          + student.getChild("plataforma").getText());
        System.out.println("Price: "
          + student.getChild("price").getText());
        System.out.println("Clasificacion: "
          + student.getChild("clasificacion").getText());
        System.out.println("Link: "
          + student.getChild("link").getText());

      }
    } catch(JDOMException e) {
      e.printStackTrace();
    } catch(IOException ioe) {
      ioe.printStackTrace();
    }
  }
}
```