

Instituto Tecnológico de Costa Rica

Administración de Tecnología de Información

Lenguajes de programación

Proyecto Programado Numero Tres

Elaborado por:

Marianne Cordero Jiménez.

Coraima Fonseca Alvarado.

Arlyn López Navarro.

Semestre I, 2014

Tabla de contenido

Resumen Ejecutivo	3
Funcionamiento	3
Propósito	4
Funcionalidades.....	5
Descripción detallada	6
Descripción del diseño de alto nivel.....	10
Librerías usadas.....	12
Problemas encontrados	14
Manual de Usuario	15
Otros detalles	28
Tecnologías usadas.....	28

Resumen Ejecutivo

La tarea programada consiste en realizar una clasificación de los ambientes del lenguaje de programación SML, los cuales son:

- **Estático:** es el ambiente en donde se hace un chequeo de tipos, guardando el tipo de la expresión y su identificador.
- **Dinámico:** es el ambiente en donde al hacer el chequeo de las expresiones en cuanto a su valor. Se almacena el identificador y valor de la misma.

Se realiza la creación de una aplicación web con Eclipse, usando principalmente el lenguaje de programación Java, además de esto es necesario el uso de las siguientes tecnologías para el desarrollo: JSP, HTML y XML. El contenido para realizar el análisis de los ambientes se encuentra de igual manera en Java.

Funcionamiento

Desde la aplicación web, el usuario deberá elegir el archivo con extensión .sml, con el propósito de mostrar la clasificación de los ambientes dinámico y estático según el contenido del mismo. Una vez elegido este archivo, primeramente, se le mostrará el contenido que hay en este. Posterior a esto, en la aplicación se le mostrarán dos opciones, las cuales son el ambiente estático y el ambiente dinámico. Si elige el ambiente estático entonces la aplicación le mostrará en formato de tabla las expresiones y el tipo de estas. Por otro lado, si elige el ambiente dinámico se le mostrará las expresiones con su respectivo valor.

Propósito

La aplicación web tiene como propósito principal analizar los entornos estático y dinámico respectivamente de un archivo “.sml”, este será cargado a la aplicación desde un computador y posteriormente recopilará la información de ambos ambientes en donde luego será retornada una tabla con el contenido del ambiente estático y otra con la información del ambiente dinámico por medio del servidor.

El análisis interno toma expresiones como if-then-else, asociaciones de variables, valores y expresiones let. Respecto al tipo de datos que se maneja, estos son booleans, integers, listas y tuplas, cada expresión será clasificada mediante la definición de una serie de métodos que recibirán cada línea del archivo (la cual contiene cada expresión) y posteriormente se completa el proceso de clasificación, de acuerdo al ambiente.

Los resultados de los entornos serán mostrados en páginas diferentes a los usuarios enlazadas desde la página principal.

Por lo tanto, para hacer posible el funcionamiento de esta aplicación, se requiere que el usuario tenga instalado Java, tener importadas ciertas librerías tanto para el manejo de los métodos que se encargan de realizar la clasificación de los ambientes como para la realización de la aplicación, así como un IDE tal como Eclipse o Netbeans para poder ejecutar la aplicación.

Funcionalidades

La aplicación posee las siguientes funcionalidades:

- **Carga del archivo:** Se elige el archivo “.SML” desde el computador que se desea someter a análisis. En este archivo deben encontrarse las diferentes expresiones a evaluar ubicada línea por línea en el mismo.
- **Mostrar el contenido del archivo:** en esta funcionalidad se muestra el contenido principal del archivo “.SML” a analizar.
- **Mostrar contenido del ambiente estático:** se muestran las variables y su respectivo tipo en formato de tabla.
- **Mostrar contenido del ambiente dinámico:** en esta funcionalidad se le mostrará al usuario el identificador y el valor en formato tabla de cada expresión evaluada.

Descripción detallada

Para realizar el análisis del ambiente estático y dinámico, se trabajó con tres clases principales, las cuales son: `NodosCompilador`, `Compilador` y `ListaSimple`.

- ***NodosCompilador***: en esta clase se construyen nodos de lista simple los cuales tienen tres campos, quienes permiten almacenar en dichos campos el identificador, tipo y valor de la expresión que se esté analizando. Esta clase permite la construcción de la lista simple.
- ***Compilador***: en esta clase se encuentran todos los métodos que se encargan de clasificar la información que va tanto en el ambiente estático como dinámico. Posteriormente se explicará con más detalle los métodos principales de esta clase.
- ***ListaSimple***: esta clase contiene métodos que permiten evaluar expresiones matemáticas, tales como sumas, restas, multiplicaciones, divisiones y elevaciones. En ella se trabaja con una lista simple, y permite crear una expresión postfija la cual facilita la evaluación de las expresiones. Al final lo que retorna es el resultado de una determinada expresión matemática.

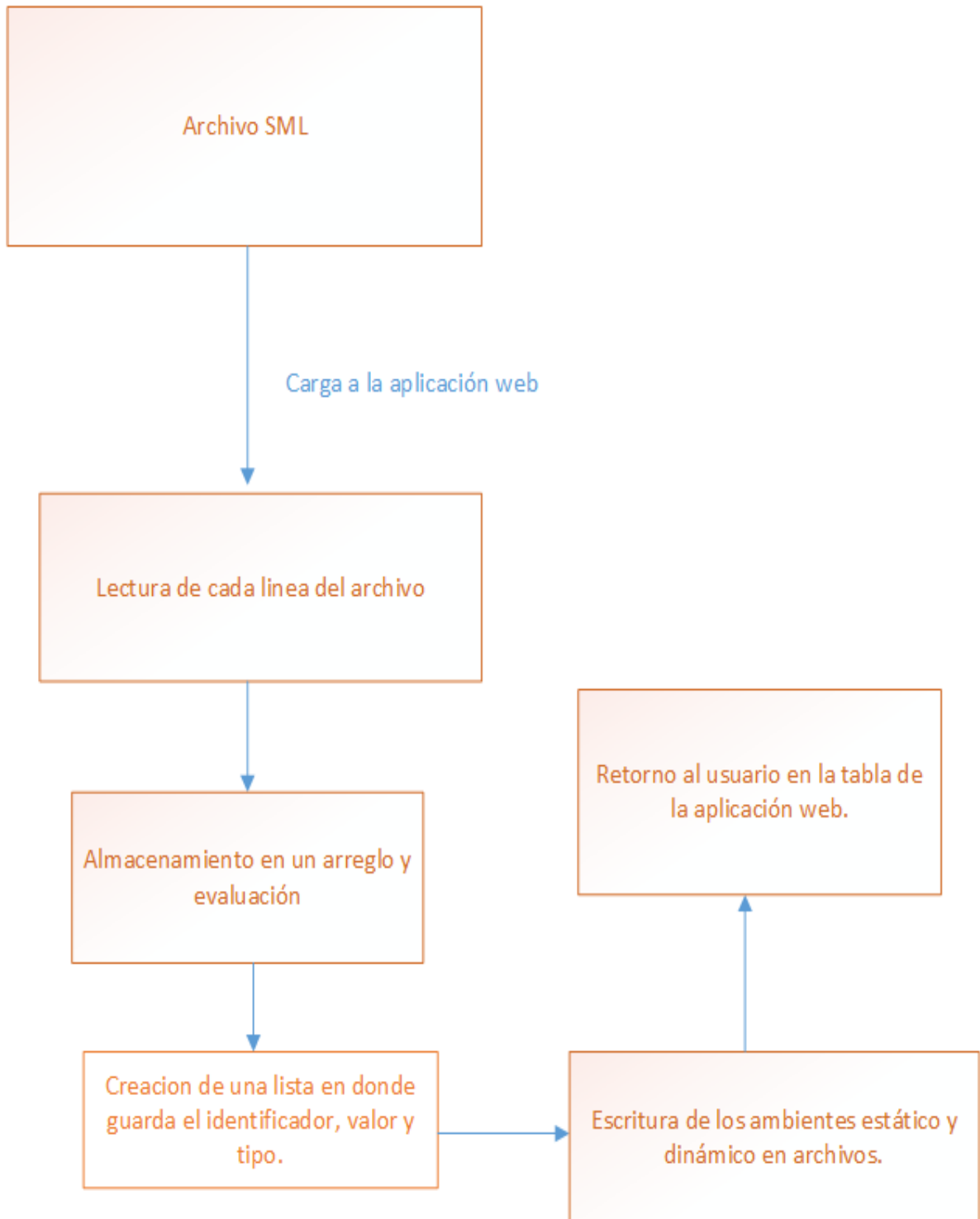
Ahora se procederá a explicar los principales métodos de la clase `compilador`, la cual se encarga básicamente de crear los ambientes mencionados anteriormente.

- ***LeerArchivo***: este método se encarga de realizar la lectura del archivo, cada línea se almacena en un arreglo, el cual posteriormente será enviado al método `evaluar_expresion` quien recibe tanto a dicho arreglo como el número de elementos que contiene el mismo, esto para que sea analizado según corresponda. Una vez que se finaliza de realizar la lectura de todo el contenido del archivo, entonces el resultado de todo el análisis se imprime, el cual está almacenado en una lista simple que se fue construyendo durante la ejecución del programa.

- **Evaluar_expresion:** este método lo que recibe es el arreglo que se creó leyendo cada línea del archivo, el cual contiene cada expresión, y también recibe el número de elementos que posee el mismo. Dependiendo de con que empiece la expresión entonces así se trabajará con un método en específico para cada una, por ejemplo si en el arreglo[0] viene la palabra “val” y el contador es cuatro entonces se llamará al método determinar_valor, el cual se encarga de procesar dicha expresión. Y así se realiza con las demás expresiones, dependiendo de la que sea, como se mencionó anteriormente.
- **determinar_valor:** este método determina el tipo de la expresión, es decir, si es un int, un bool, una lista, una tupla o un string. Recibe el identificador de la expresión y el valor de la misma, donde en dicho método lo que se procesará será el valor. Una vez que se finaliza el análisis del tipo de dato, se almacena en la lista principal el tipo y valor de la expresión.
- **determinar_valor_tupla:** en este método se determina el tipo de dato de cada uno de los elementos de la tupla, ya sean int, strings, otras tuplas, listas o booleans. Básicamente lo que se hace es evaluar cada dato y cada vez que se encuentre una “,” entonces esta será reemplazado por un “*” para mostrar por medio de esos el tipo de dato de cada uno de sus elementos.
- **tipos_lista:** se determina el tipo de dato que corresponde cierta lista, ya sea una int list, bool list, string list, etc. En este método se valida que las listas sean homogéneas, es decir, que el tipo de todos los elementos que posee sean iguales; de lo contrario, se le indicará al usuario que no lo son.
- **Let_it_be:** evalúa las expresiones let. Básicamente lo que hace es analizar el tipo y el valor de la variable que se define en el let, es decir, si viene let val z=5 entonces primeramente analiza el val=5. Posterior a esto analiza la expresión definida en el in.

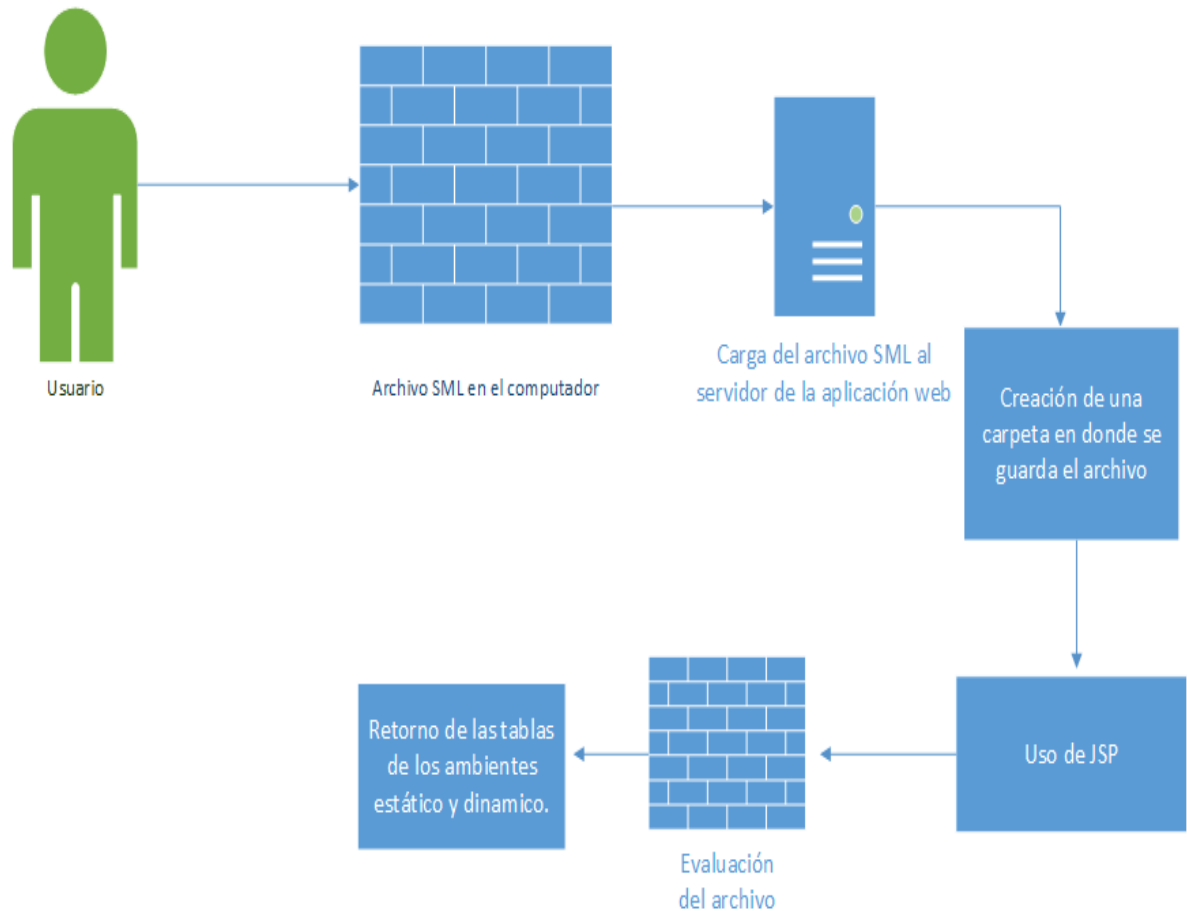
- **Evaluar_if:** este método lo que hace primeramente es evaluar el if-then-else, es decir, si es un true o un false, dependiendo de si se cumple o no la condición que se indica. Posterior a esto se analiza la expresión que está en el then y por último la del else.
- **ambiente_estatico:** este método realiza la escritura en el archivo de los datos que se recopilaron en el ambiente estático, es decir, el identificador y tipo de la variable. Esto para después cargar dicho archivo desde la página y mostrar su contenido en la tabla.
- **ambiente_dinamico:** al igual que el anterior, este método realiza la escritura en el archivo de los datos que se recopilaron en el ambiente dinámico, es decir, el identificador y el valor de la expresión. Esto para después cargar dicho archivo desde la página y mostrar su contenido en la tabla.
- **Es_Numero:** en esta función se inicializa una variable resultado con el valor booleano de true tomara desde diferentes bloques de código el Objeto valor, este propiamente de la variable y trata de hacer un cast a int, en caso de que se logre retornará el true, de lo contrario retornará falso.

Una manera gráfica de presentar la lógica general de la clase compilador es la siguiente:



Descripción del diseño de alto nivel

Para hacer la descripción de alto nivel basándose en la arquitectura de la aplicación se ha hecho varios diagramas ilustrativos que muestra cómo se maneja de manera sencilla la aplicación por dentro.



Respecto a los principales métodos de los cuáles se conforma la aplicación, es decir, para elaborar en si la página, están:

- **CargarArchivo:** esta funcionalidad hace la carga del archivo “.SML” y lo guarda en una carpeta dentro de la aplicación web, dicho formulario podría ser enviado a “Multipart” en donde se hace una revisión del archivo sobre algunos punteros, se crea un FileUpload, se procesa una solicitud para crear un ArrayList y una lista en caso de que exista derivación de ficheros.

Posteriormente se recorre una lista, se rescata el archivo y se hace la comprobación de que este sea un Fichero por su parte en caso de que no lo sea se obtendrá toda la información del archivo, lee cada una y las separa cuando las encuentra, ingresa cada token en un arreglo y hace las llamadas necesarias a la parte lógica.

- **DoGet:** realiza la obtención de los datos que contiene el archivo.
- **DoPost:** es un método que permite retornar los resultados a la pantalla.
- **Ambiente_dinámico:** cuando termina de hacer la carga, lectura y análisis del archivo esta funcionalidad toma todos los datos clasificados anteriormente y los escribe en un archivo.
- **Ambiente_estático:** cuando termina de hacer la carga, lectura y análisis del archivo esta funcionalidad toma todos los datos clasificados anteriormente y los escribe en un archivo, (identificador y tipo de variable).

Librerías usadas

A continuación se hará la definición de las importaciones que se usan en la tarea programada, las cuales permiten realizar distintas operaciones con los datos.

<pre>import java.io.BufferedReader; import java.io.BufferedWriter; import java.io.File; import java.io.FileNotFoundException; import java.io.FileOutputStream; import java.io.FileReader; import java.io.FileWriter; import java.io.IOException; import java.io.OutputStreamWriter; import java.io.PrintWriter; import java.util.ArrayList; import java.util.Iterator; import java.util.List; import java.util.StringTokenizer;</pre>	<p>Lector de consola. Escribe en consola. Acepta archivos. Informa de si no existe una ruta definida. Permite la escritura en distintas plataformas. Lee archivos. Escribe archivos. Alerta sobre interrupciones en entrada o salida. Puente entre bytes y datos. Imprime datos. Genera arreglos de datos. Permite la iteración. Genera listas. Permite hacer la tokenización de datos.</p>
<pre>import javax.servlet.ServletException; import javax.servlet.http.HttpServlet; import javax.servlet.http.HttpServletRequest; import javax.servlet.http.HttpServletResponse;</pre>	<p>Todas estas importaciones sirven para hacer el servidor de la aplicación web, por ejemplo para el manejo de errores, las respuestas que enviara el mismo así como las llamadas que se hacen.</p>

```
import org.apache.tomcat.util.http.fileupload.FileItem;  
import org.apache.tomcat.util.http.fileupload.FileUploadException;  
import org.apache.tomcat.util.http.fileupload.disk.DiskFileItemFactory;  
import org.apache.tomcat.util.http.fileupload.servlet.ServletFileUpload;  
import org.apache.tomcat.util.http.fileupload.servlet.ServletRequestContext;
```

Estas clases pertenecen a la librería JSP, en donde se crea un ambiente en conjunto con el servidor para el funcionamiento del servidor.

Posterior a entender el diseño de alto nivel y las funcionalidades de la aplicación será necesario proceder al funcionamiento en conjunto de todas las partes que la conforman. En primer lugar se hace la creación de la aplicación web, está por su parte está compuesta por un servidor, entonces se procede a cargar un archivo de tipo SML, por medio del botón se hace la selección del archivo.

El archivo será analizado, este primeramente pasará por una fase de comprobación, sabiendo que no es un fichero, sino un archivo, posteriormente se hará la lectura línea por línea, la separación de cada “token” será almacenada en un arreglo. Por cada línea se hará la evaluación con los distintos métodos lógicos que se han establecido para determinar valores y tipos de variable.

Se procederá a realizar la escritura de los resultados mediante las funcionalidades de los ambientes estáticos y dinámicos, en donde se utilizaran dichos datos para generar las tablas de análisis que serán devueltas al usuario.

Problemas encontrados

En el momento que se trató de crear la aplicación después de haberse subido el archivo persistió un problema con los métodos get para crear las tablas, ya que si se hacía un ciclo, este repetitivamente insertaría casillas a la tabla, sin embargo se encontró una solución implementando una condición de parada por líneas leídas del archivo.

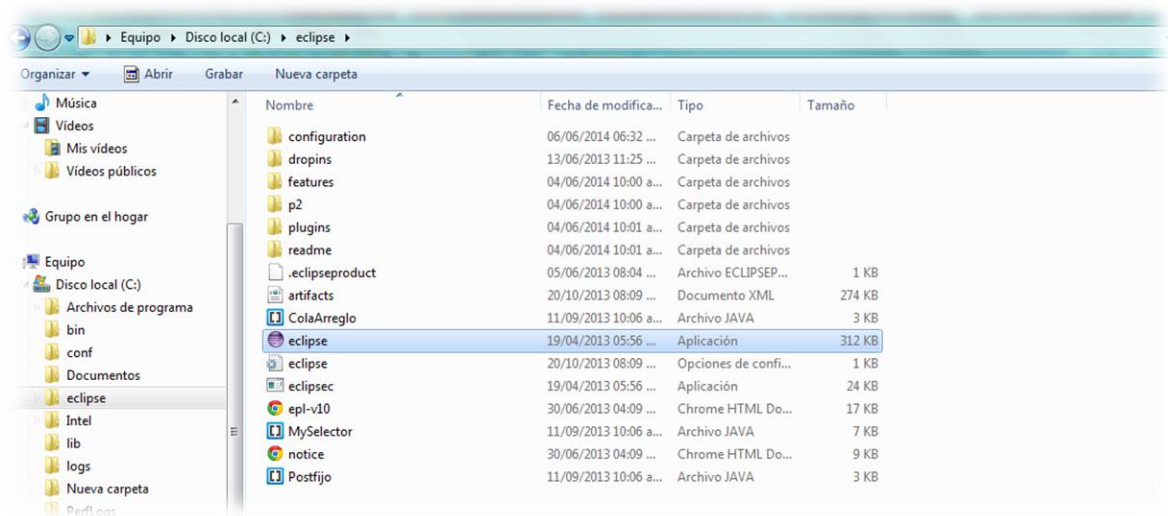
En la funcionalidad del Let, este no trataba las expresiones complejas, solo tomaba el primer elemento y este no aplicaba la operación matemática por lo que fue necesario implementar la clase ListaSimple para tomar los demás valores y retornar un acumulado.

Manual de Usuario

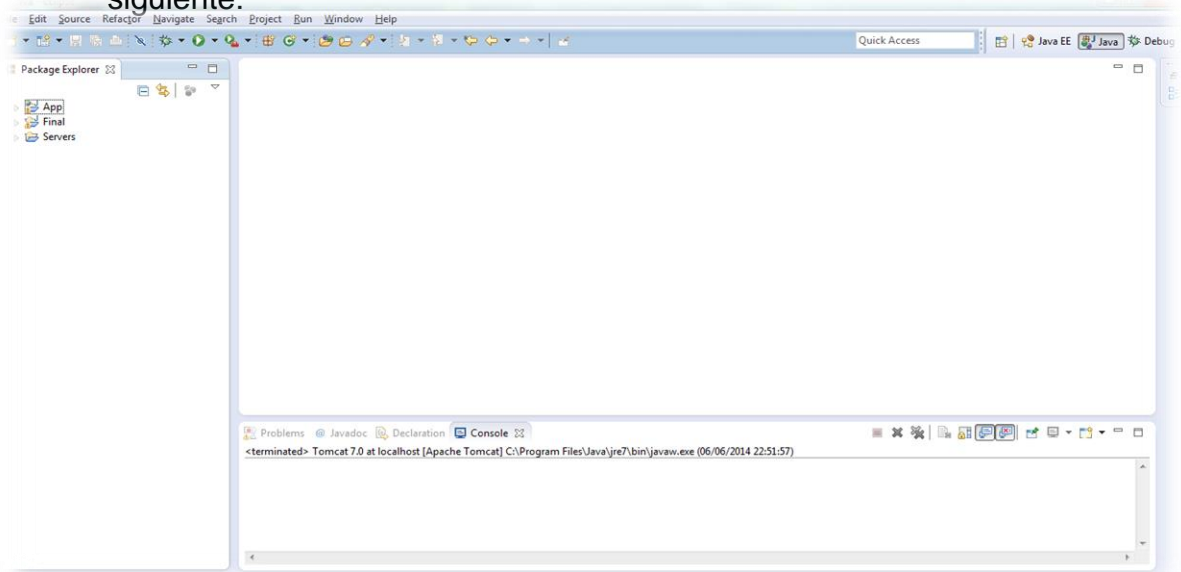
Para la utilización de la aplicación web, se necesitará de la previa instalación en su computador de eclipse (versión Kepler) y el servidor web Apache Tomcat.

La instalación de eclipse se puede hacer de la siguiente forma:

1. Descargar eclipse de la siguiente dirección:
<https://www.eclipse.org/downloads/>.
2. Una vez descargado se procesa a su instalación el cual solo se ejecutará de esta forma:



3. Al ejecutarlo ya estará listo para ser utilizado y se verá algo como lo siguiente:

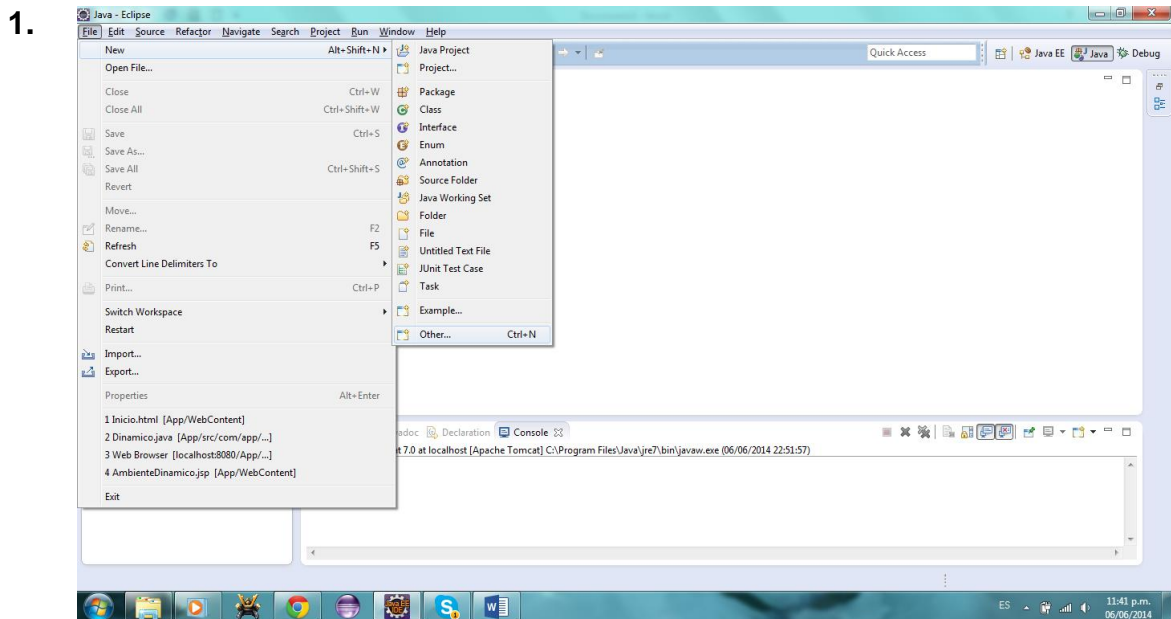


La instalación de Apache Tomcat se puede hacer de la siguiente forma:

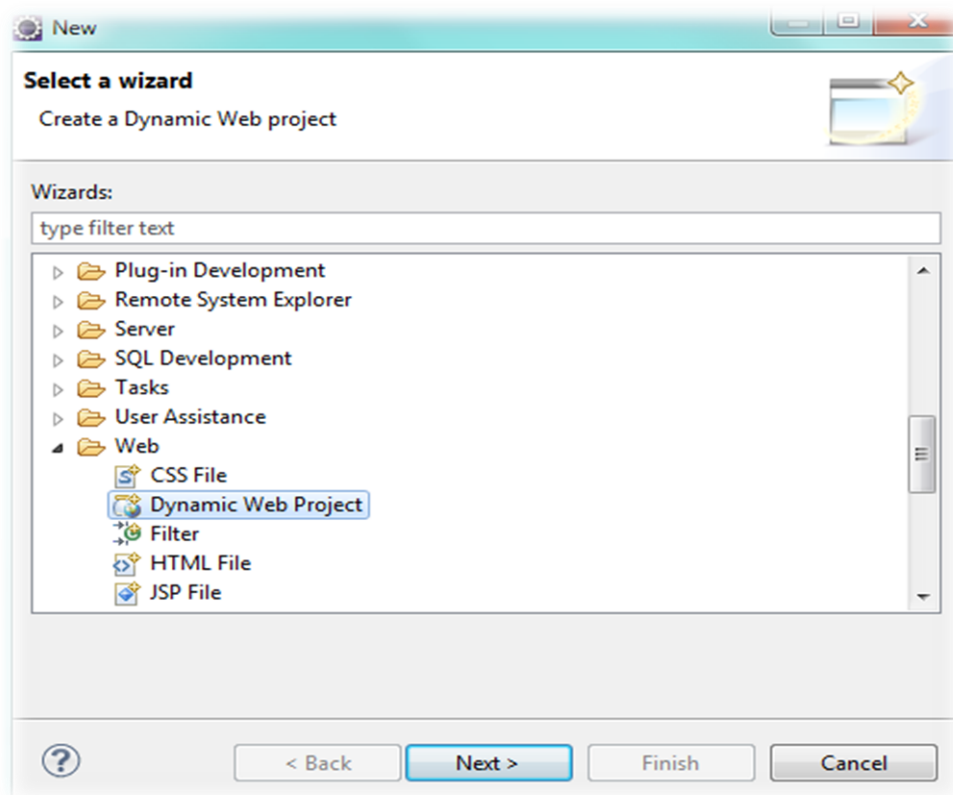
1. Descargar la versión más reciente de Apache Tomcat(Tomcat 7.0) de la siguiente dirección: <http://tomcat.apache.org/download-70.cgi>
2. Una vez descargado se deberá instalar en su equipo.



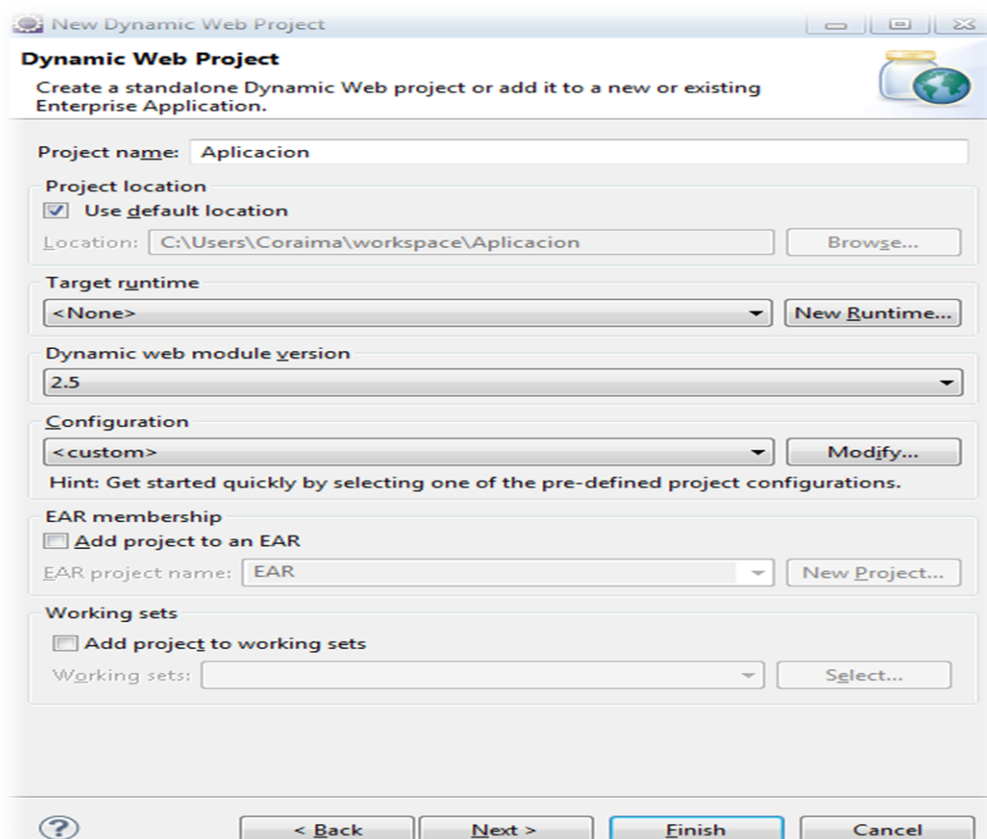
Cuando se hayan descargado las herramientas necesarias para la utilización del App se deberá crear un Dinamic Web Project en Eclipse siguiendo los siguientes pasos:



2.

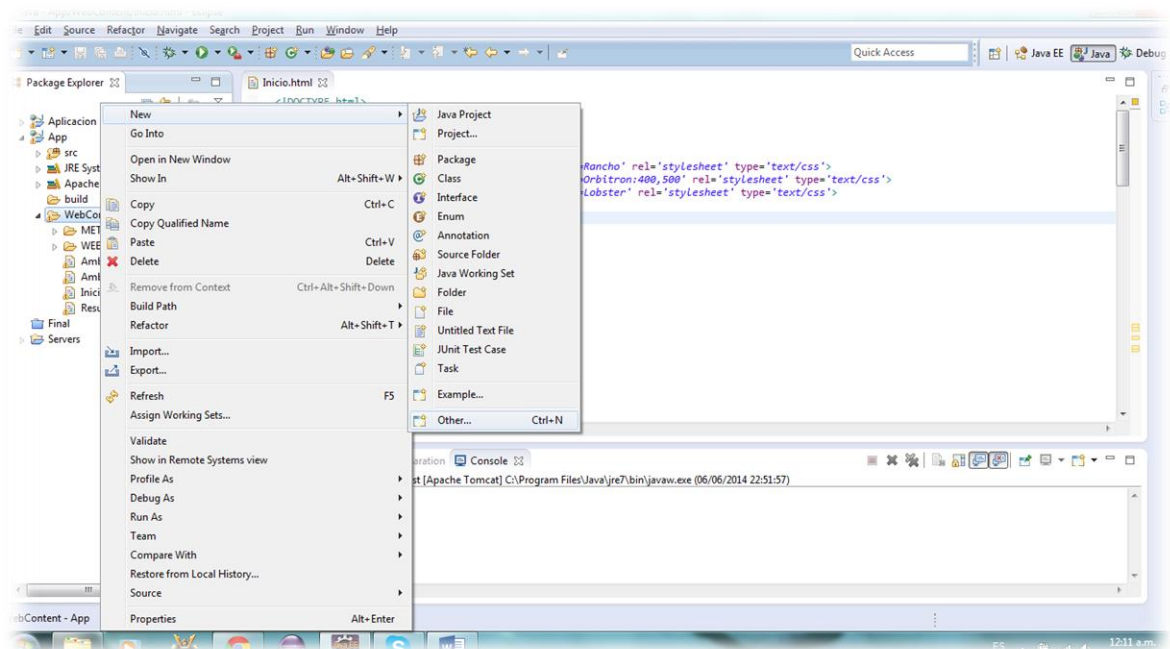


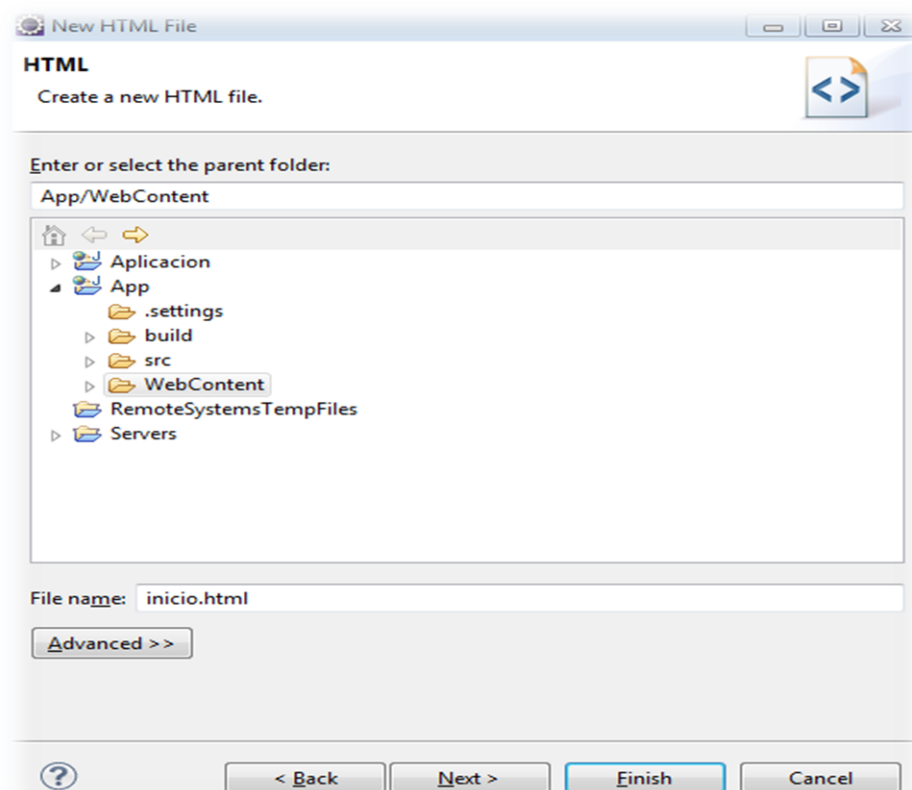
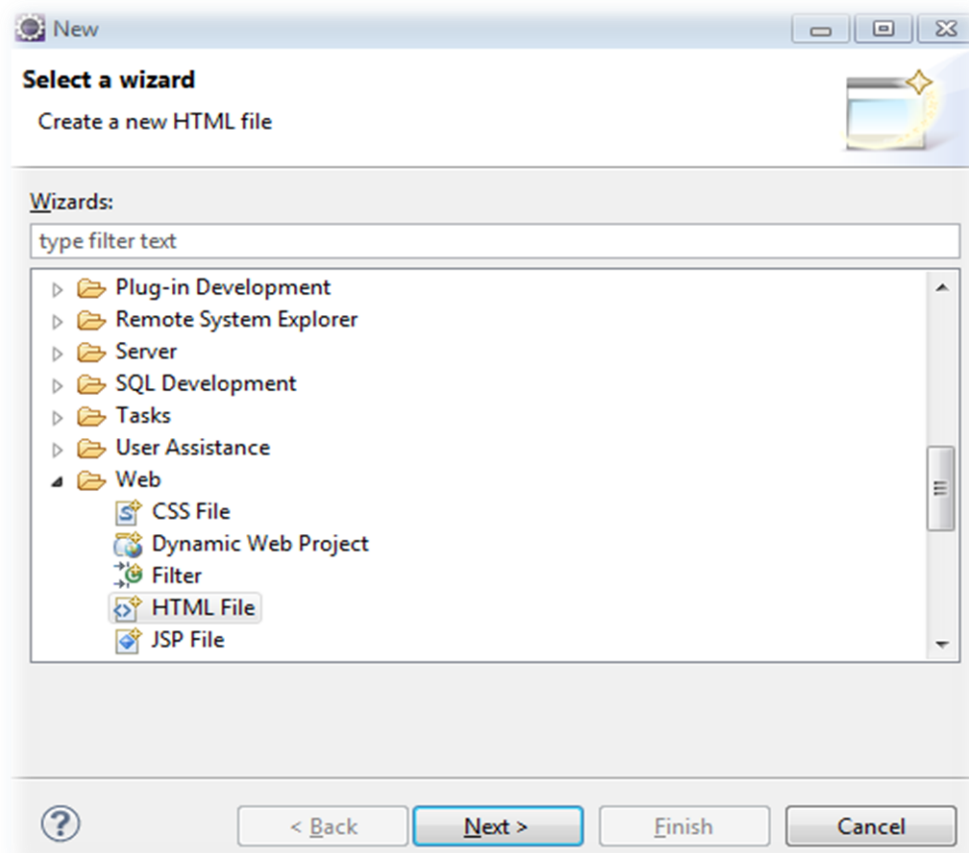
3.



The screenshot displays the Eclipse IDE environment. The Package Explorer on the left side of the IDE shows a project named 'Aplicacion'. Under this project, there are four folders: 'src', 'JRE System Library [J2SE-1.5]', 'build', and 'WebContent'. The main editor area is currently empty. The bottom console window is active, showing a message that indicates the termination of Tomcat 7.0 at localhost, with the path 'C:\Program Files\Java\jre7\bin\javaw.exe' and the timestamp '(06/06/2014 22:51:57)'.

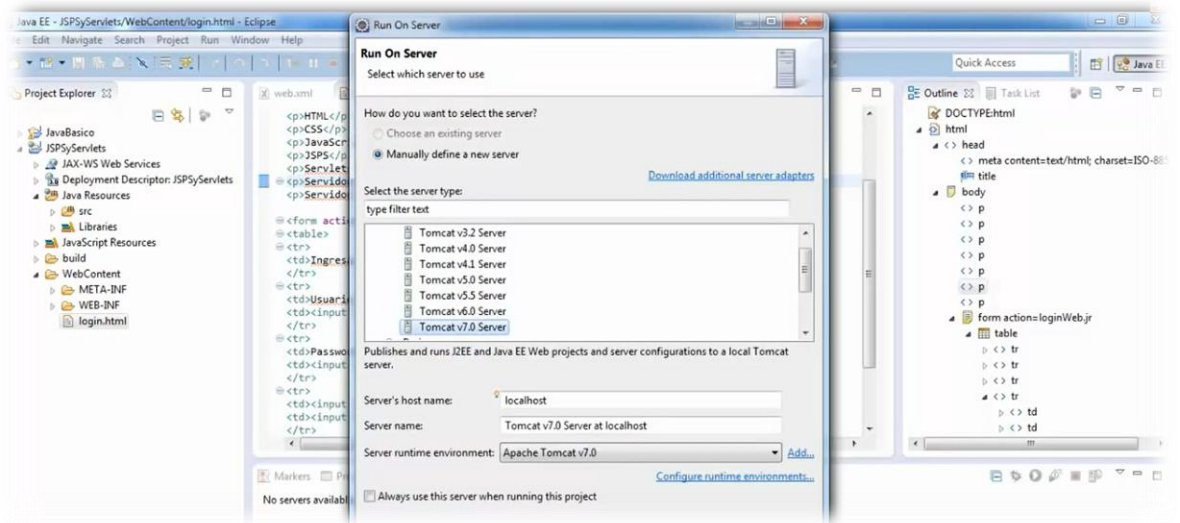
1. Creación de una página HTML.





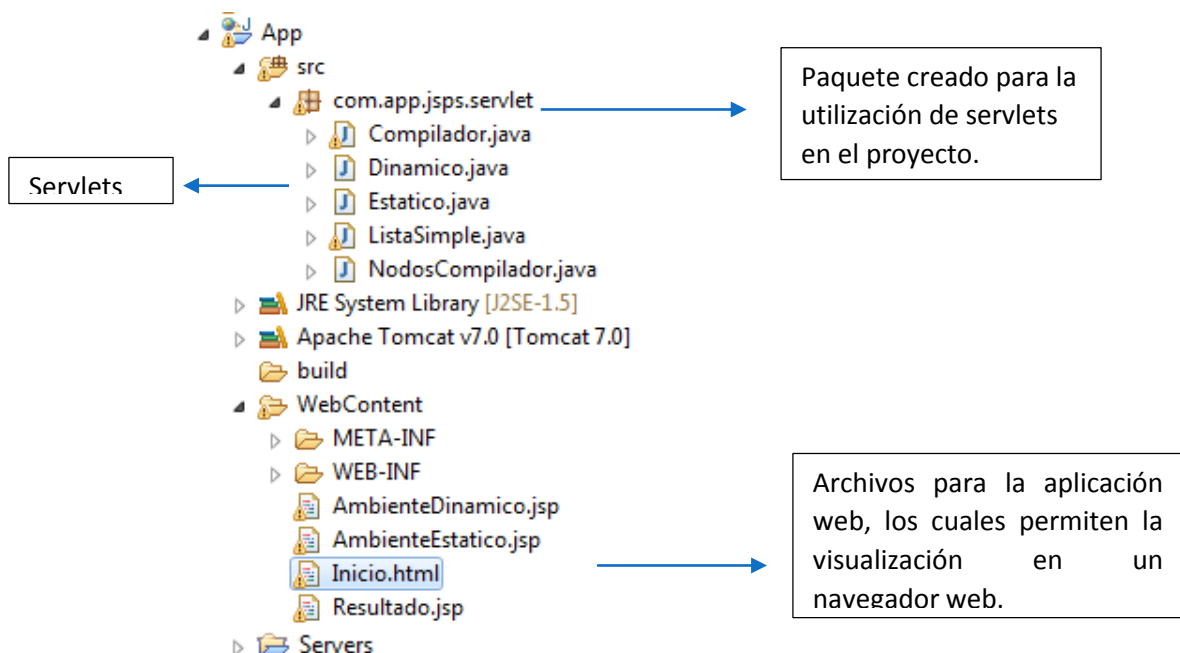
Se deberá dar un clic en **Finish** y se creará la página HTML. Posteriormente se deberá poner en funcionamiento el servidor:

1.

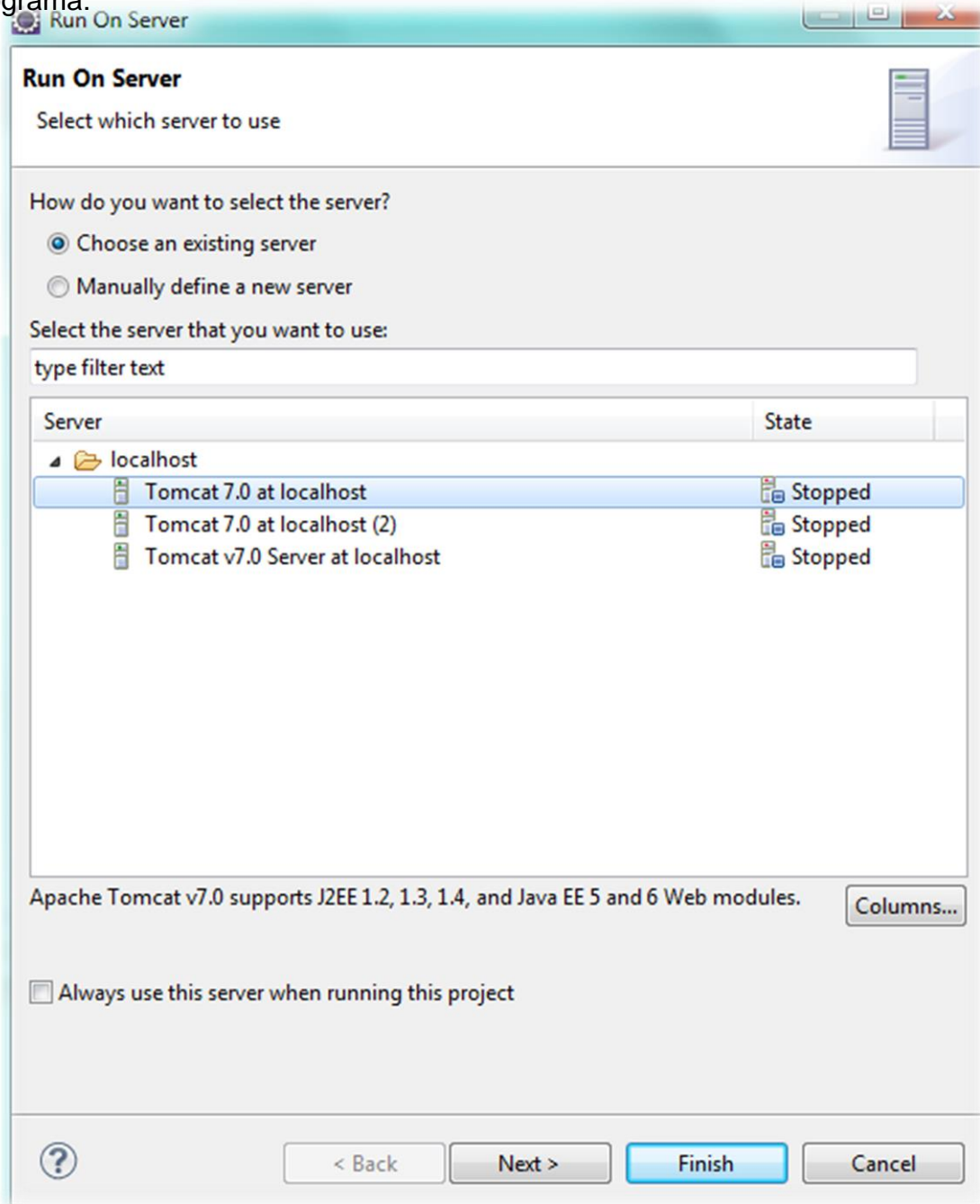


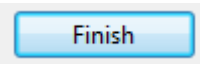
2. Se deberá dar un clic en **Finish** y estará en funcionamiento el servidor para ser utilizado.

Para poder ejecutar la aplicación se deberá tener dentro del proyecto creado al inicio del manual los siguientes archivos y su estructura será parecida a la mostrada a continuación:

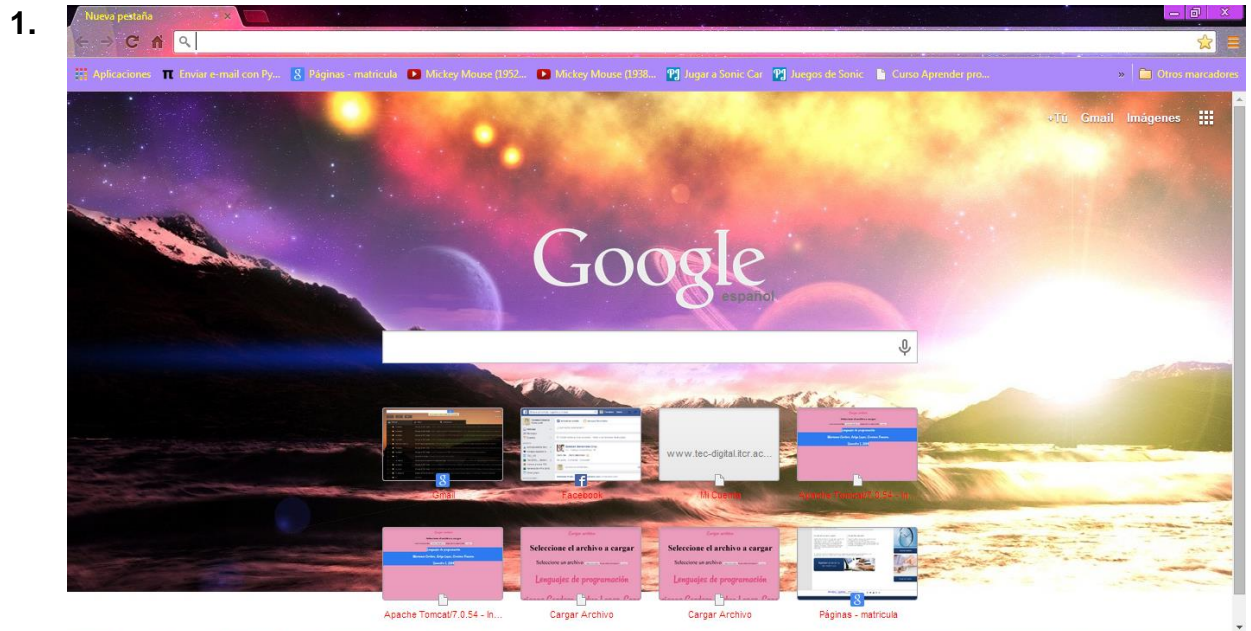


Cuando se haya creado el proyecto de manera correcta, se ejecutará el programa:

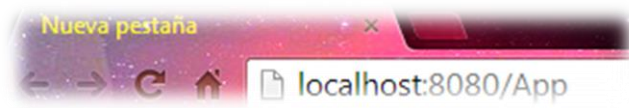


Se deberá dar un clic en  y estará en funcionamiento el servidor para ser utilizado.

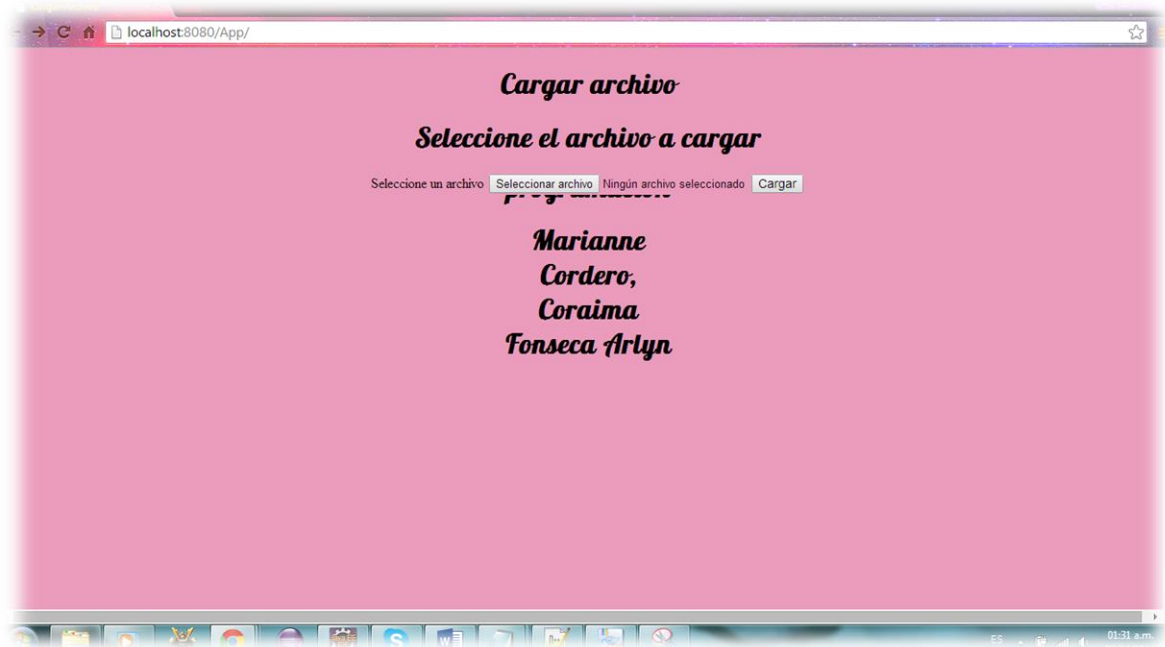
Una vez ejecutado el programa se procederá a abrir el navegador web, como se muestra:



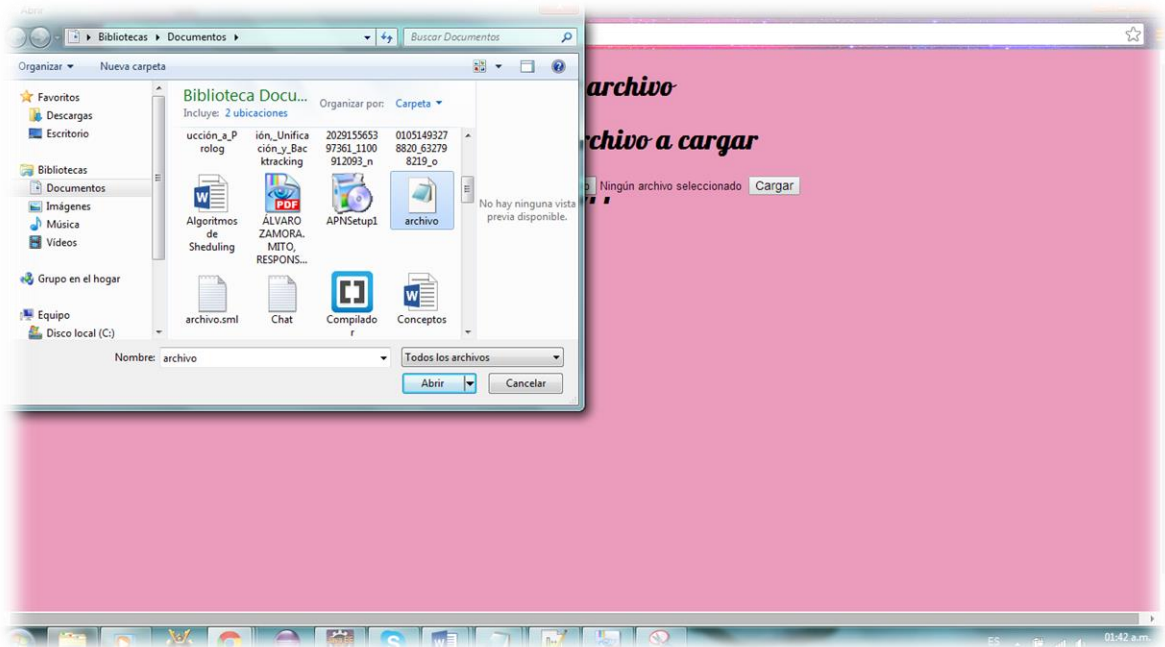
2. Poner en el navegador la siguiente dirección:




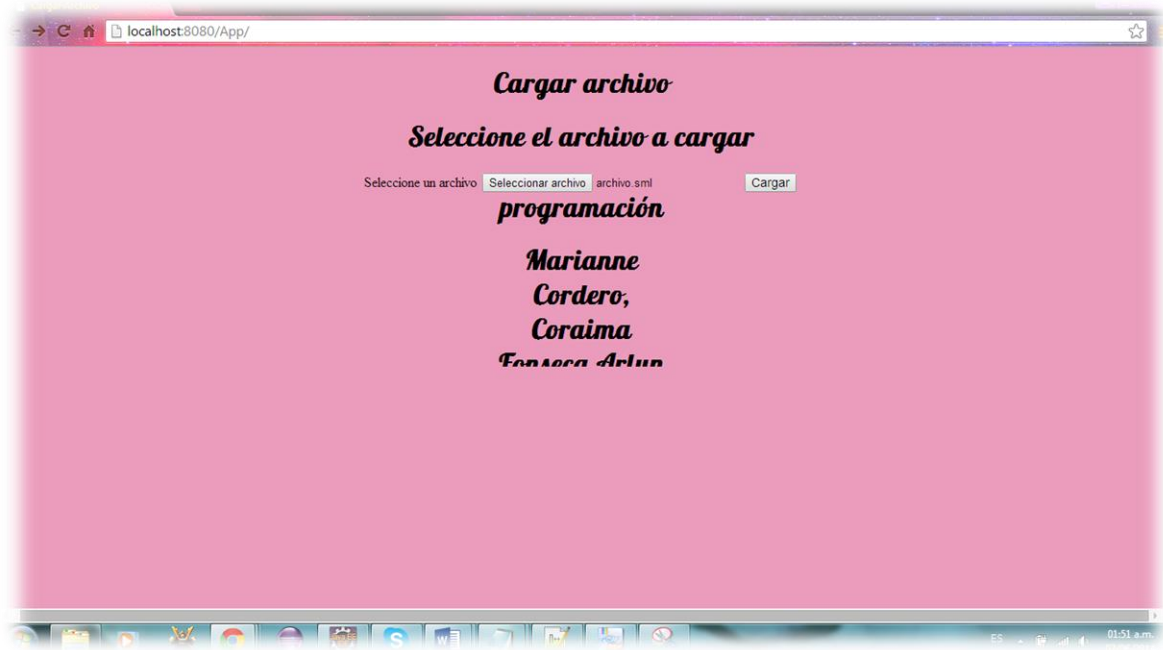
3. Se mostrará una página como la siguiente:

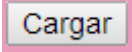


4. El usuario debe escoger el archivo a cargar, en este caso será un archivo cuya extensión es .sml como se muestra a continuación:



5. Cuando el dan  al archivo, este será cargado por la aplicación.



6. Posteriormente se deberá presionar el botón de  y se redireccionará a otra página donde se mostrará información relacionada con el archivo cargado, entre la información a mostrar se encontrará:

- Nombre del archivo
- Tipo de contenido del archivo.
- Tamaño del archivo.
- Contenido del Archivo.

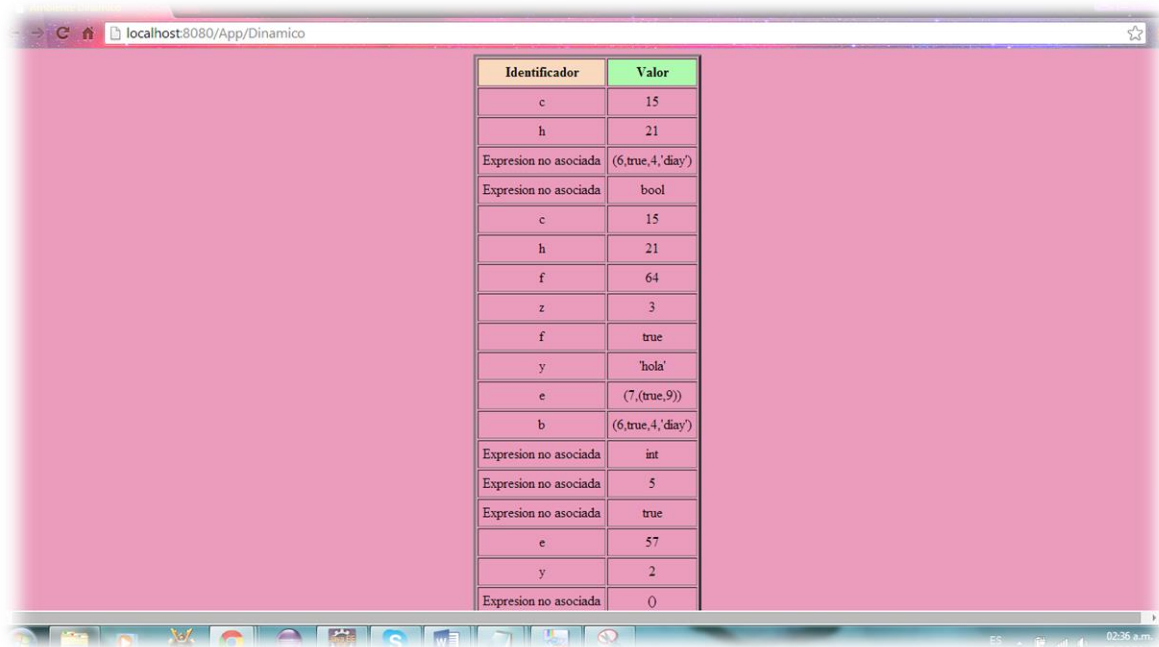


7. La página anterior muestra dos botones uno que dice Ambiente Estático y el otro Ambiente Dinámico.

Si se presiona el botón **Ambiente Estatico** se redireccionará a una página en donde se mostrará una tabla con el identificador y el tipo.



Si de lo contrario se presiona el botón **Ambiente Dinamico** se re direccionará a una página en donde se mostrará una tabla con el identificador y el valor.



The screenshot shows a web browser window with the address bar displaying 'localhost:8080/App/Dinamico'. The main content area features a table with two columns: 'Identificador' and 'Valor'. The table contains 20 rows of data, including identifiers like 'c', 'h', 'f', 'z', 'y', 'e', 'b' and values like '15', '21', '64', '3', 'true', 'hola', '(7,(true,9))', and '(6,true,4,'day')'. Some rows are labeled 'Expresion no asociada'.

Identificador	Valor
c	15
h	21
Expresion no asociada	(6,true,4,'day')
Expresion no asociada	bool
c	15
h	21
f	64
z	3
f	true
y	'hola'
e	(7,(true,9))
b	(6,true,4,'day')
Expresion no asociada	int
Expresion no asociada	5
Expresion no asociada	true
e	57
y	2
Expresion no asociada	0

Así finalizará la correcta utilización de la aplicación.

Otros detalles

Tecnologías usadas:

- **JSP:** Java Server Pages brinda la posibilidad de crear páginas web dinámicas basándose en HTML y XML. Es necesario correr el servidor bajo la tecnología de Apache.

El código es compilado como cualquier clase Java y es multiplataforma por lo que si se deseara realizar cambios no presentan muchos problemas.

- **HTML:** lenguaje de marcas de hipertexto, sirve para hacer la parte grafica de la aplicación como por ejemplo cambiar el color de letra, tamaños, importar imágenes entre otras cosas.
- **XML:** Lenguaje de marcas extensible, hace referencia a un lenguaje de programación para realzar un almacenamiento de datos de forma legible, especializado para realizar una óptima estructuración de datos en los documentos. Ayuda a crear hojas de cálculo, entre otros documentos.
- **Apache:** Es una tecnología para hacer servidores web con el protocolo HTTP, con él es posible realizar configuración de contenido web tanto como en bases de datos.