Dapena Gómez Jesús

UAH

Practica 5. Parte 1

Memoria practica5. Parte 1

Contenido

[Preparación del entorno 1](#_Toc466559234)

[Hola Mundo 3](#_Toc466559235)

[Calculadora 3](#_Toc466559236)

[Formularios 3](#_Toc466559237)

[Primitiva 3](#_Toc466559238)

[init() 3](#_Toc466559239)

[comprueba() 4](#_Toc466559240)

[service() 4](#_Toc466559241)

[Trabajo con datos 5](#_Toc466559242)

[Init() 8](#_Toc466559243)

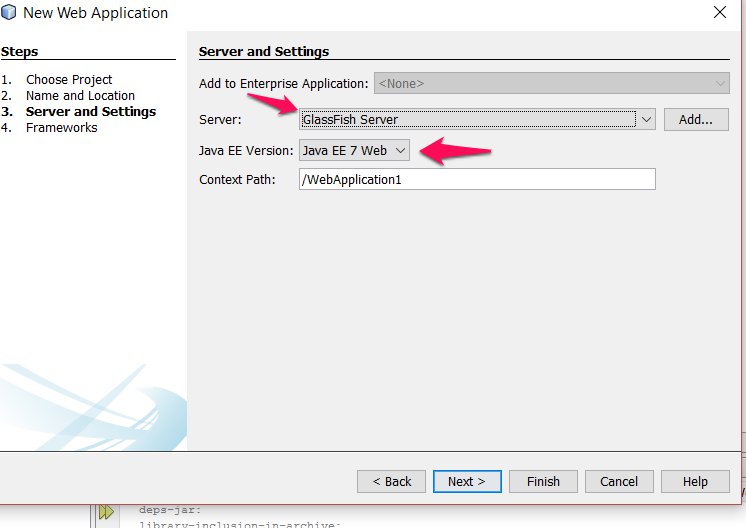
[Service() 8](#_Toc466559244)

[destroy() 8](#_Toc466559245)

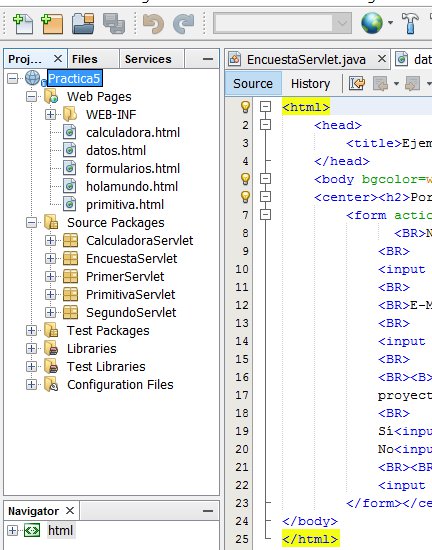
# Preparación del entorno

Para poder realizar la práctica descargamos la última versión de Netbeans compatible para trabajar con Java EE.

Una vez descargado nos dispusimos a descargar el servidor que se encargará de mantener las aplicaciones realizadas, en nuestro caso, glass fish.



Una vez teníamos todo el entorno preparado simplemente teníamos que copiar el código proporcionado en la práctica con algunas modificaciones que se detallarán en cada apartado.



# Hola Mundo

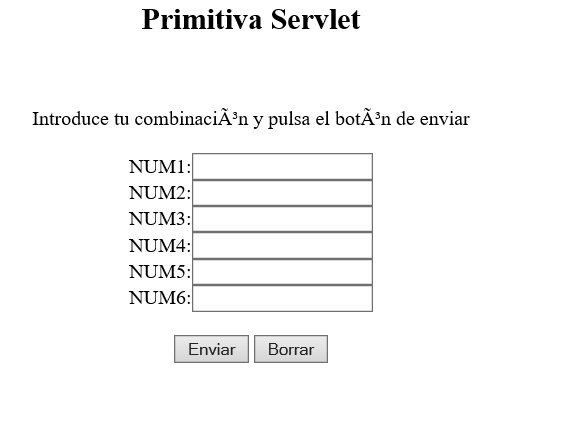
# Calculadora

# Formularios

# Primitiva

El código html de inicio se encargará de mostrar una página donde aparecerán los campos necesarios para que el usuario rellene con su propia apuesta.

A parte de esto se muestra el botón de enviar que realiza la función submit y el botón borrar con la función de reset.



En nuestro caso el código del servlet lo hemos añadido dentro del paquete PrimitivaServlet, por lo que para ser llamado desde primitiva.html hemos tenido que añadir lo siguiente:

C:\Users\jesus\AppData\Local\Skitch\Captura_de_pantalla_111016_040659_PM.jpg

El código del servlet por otro tiene las siguientes funciones importantes:

## init()

Es la función que es llamada cuando se carga el servidor. Se encargará de generar el número de la primitiva que ha resultado ganador.

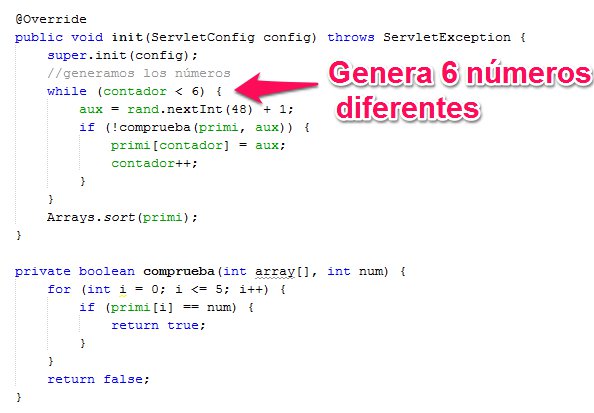
Hasta que no se tengan 6 números diferentes en el array irá generando nuevos números entre el 1 y el 48.

Para comprobar que todos los números son diferentes llama a la función comprueba.

## comprueba()

Devuelve un booleano, comprueba si el número pasado como parámetro se encuentra en el array también pasado como parámetro.

Es utilizado por la función init() para asegurarse de que todos los números son diferentes.



## service()

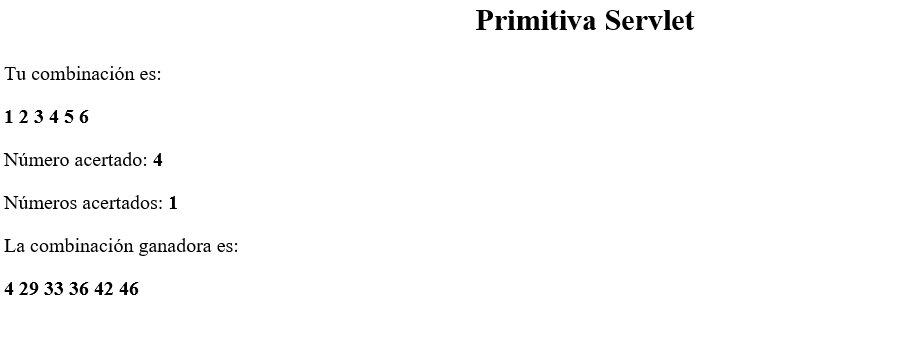
Es la función encargada de responder al cliente, una vez recibida una petición este se encarga de leer los números del usuario.

Una vez leídos y mostrados a través de la función de búsqueda binaria proporcionada por jaba.util.Arrays se encargará de comprobar cada cifra del usuario con la combinación ganadora, obteniendo de esta forma el número de aciertos del usuario.

Finalmente se muestra el número de aciertos y el de la combinación ganadora.



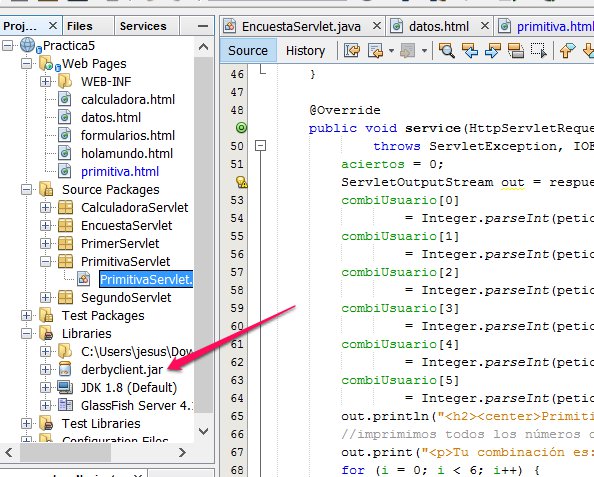
Finalmente la aplicación queda de la siguiente forma cuando se ejecuta:



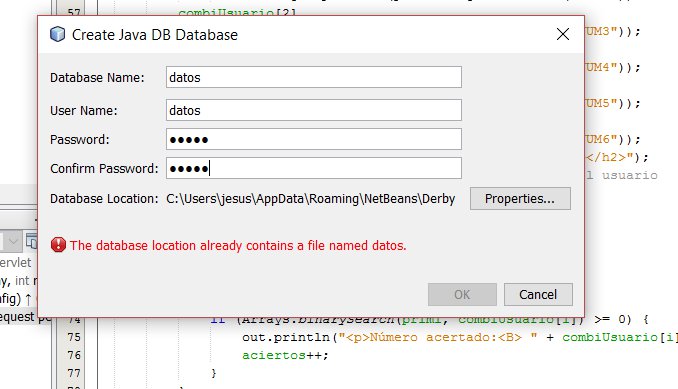
# Trabajo con datos

En este caso lo primero que teníamos que hacer era configurar el servidor derby y la base de datos que se iba a utilizar.

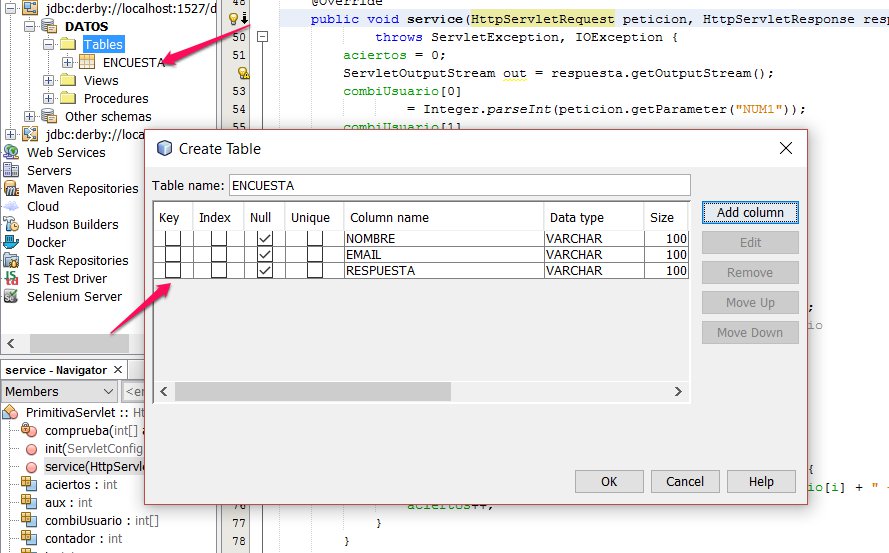
Para eso descargamos la librería derby desde la página correspondiente y la insertamos en el proyecto:



Una vez añadida en el proyecto en servicios sobre Java DB con el botón derecho seleccionábamos crear una nueva base de datos. En nuestro caso la base de datos se llamaba datos, con el nombre de usuario y contraseña también como datos para simplificar el proceso:



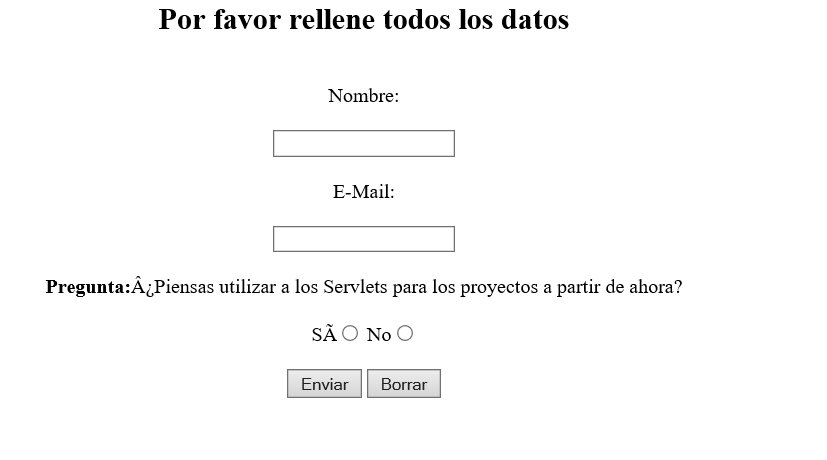
Una vez creada la base de datos nos conectamos a ella y el paquete de tablas con el botón derecho creamos una nueva tabla:



Una vez explicada la instalación de derby vamos a ver como es el código html.

Es un html muy sencillo en el que únicamente se muestra dos campos de texto para el usuario y el email y dos radioButton para indicar si la respuesta es SI o NO.

Por supuesto al final tenemos los botones de enviar la encuesta o de borrar todos los datos. La página es como se muestra a continuación:



Vayamos entonces a explicar el código del servlet. En este caso hay dos variables globales llamadas mandato y conexión que se encargarán como su nombre indica de gestionar todas las conexiones con la base de datos.

Las funciones importantes son init(), service() y destroy().

## Init()

En este caso se intentará realizar la conexión a la base de datos creada anteriormente. Para que funcione de manera correcta hemos tenido que modificar ligeramente el código:

* url pasa a ser la dirección local donde se encuentra la base de datos.
* usuario y contraseña pasa a ser los utilizados en la base de datos.
* Class for name ahora usará org.apache.derby.jdbc.ClientDriver.

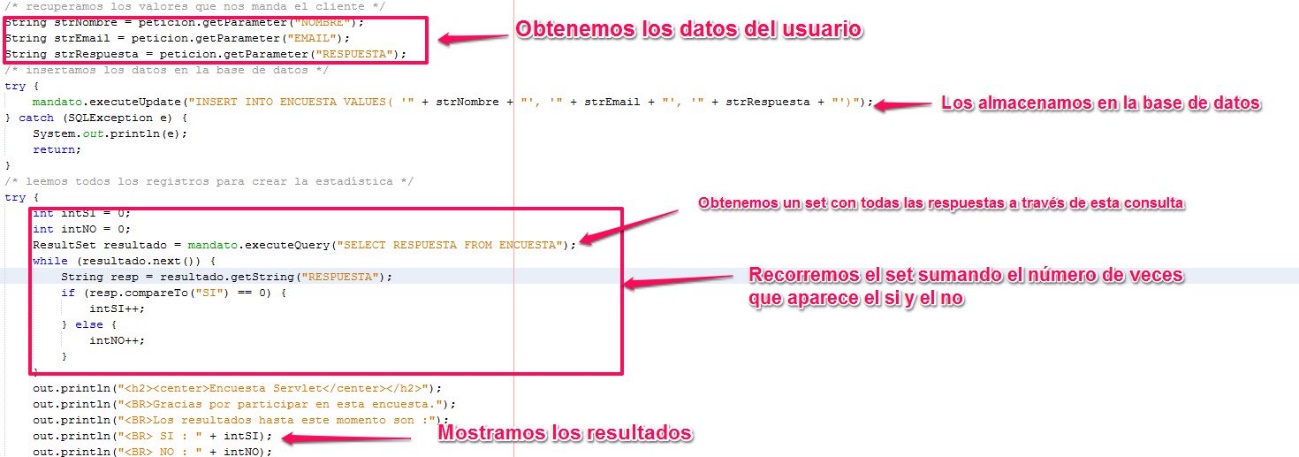
Una vez realizados estos cambios el código queda de la siguiente forma:



## Service()

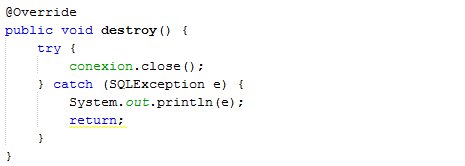
Al igual que en los casos anteriores es la encargada de proporcionar el servicio. En este caso una vez tenemos los datos introducidos por el usuario realizaremos una consulta en la bbdd donde se guardarán los datos del usuario.

Una vez guardados los datos entonces se realiza otra consulta, en este caso para obtener una lista con todas las respuestas de los usuarios, contando en dos variables el número de síes y de noes.



## destroy()

La función destroy se encargará simplemente de que cuando se cierre la conexión con el cliente cerrar también la conexión con la base de datos.



Finalmente la aplicación ejecutándose queda de la siguiente manera:

