**Trabajando con el archivo binding.xml**

Cuando una pagina web dentro de Canis es solicitada se utiliza el archivo binding.xml para obtener toda la logica de renderizado necesaria para mostrar la página, para entender esto mejor podemos partir de un ejemplo de archivo binding.xml:

<?xml version="1.0"?>

<vistas>

<vista **nombre=’inicio’** **url='inicio.php'** **clase=’InicioDelegate’** **template=”default”** **styles=”proyecto/inicio.css”**>

**<consulta destino=’bienvenida’>getBienvenida</consulta>**

</vista>

</vistas>

En el archivo mostrado se puede apreciar las definiciones ejemplo de vistas de un modulo desarrollado con Canis, en particular en este ejemplo, se esta definiendo la vista de cuyo **URL es “inicio.php**” la ruta especificada en esta propiedad viene dada a partir del directorio “/view”del frameword, es decir, si trataramos de acceder a esa vista desde el directorio raiz del proyecto su ruta seria /view/inicio.php, pero como todas las vistas deben ir dentro del directorio “/view” entonces Canis inicia la busqueda del archivo a partir de este directorio.

A esa vista se le ha dado el nombre de “**inicio**” dentro del **enrutamiento logico del proyecto** (Canis asigna nombres logicos a todas las vistas, de esta manera se evita cambiar todos los enlaces de la pagina si el nombre del archivo php cambia, por otro lado logramos darle nombres mucho mas amigables al usuario y optimizados para la indexacion de los buscadores).

Ademas, se le ha asignado la **clase InicioDelegate.php** (Delegate) para las operaciones CRUD dentro de la vista y se ha definido especificamente un metodo de consulta llamado “**getBienvenida**” cuyo resultado es esperado dentro de una variable PHP llamada **$vars[“bienvenida”]**.

Entonces, ¿Que quiere decir todo esto? Bueno ahora vamos a ir explicando con mas detalle todas y cada una de estas propiedades.

* **Nombre**:

Representa un String unico entre todas las vistas del proyecto que permitira realizar el enrutamiento de las paginas sin la necesidad de verse atados al nombre fisico de la vista, esto nos trae como principal ventaja que podemos cambiar el nombre del archivo de la vista sin afectar casi en ninguna manera los vinculos de la pagina.

* **URL**

Es el nombre fisico de la vista, en nuestro caso “inicio.php”. Todas las vistas deben estar ubicadas en el directorio “view” del proyecto. Canis buscara las vistas de este directorio en adelante, pudiesemos por ejemplo crear un directorio “usuario” dentro de la carpeta “vista” y si pusieramos una vista cuyo fichero es llamado “login.php” ahi dentro, necesitariamos definirla en bindings.xml de la siguiente manera “usuario/login.php”

* **Clase**

Es la clase encargada de realizar las peticiones, inserciones y modificaciones de datos contra el servidor, se utiliza este patron de diseño para enrutar dinamicamente las llamadas a los metodos segun la configuracion.

Este tipo de planteamiento tambien brindar una mayor organizacion del proyecto al amuentar el nivel de granularidad de la informacion y realizar un enfoque transaccional.

* **Template**: Este parámetro es opcional, en caso de no ser definido Canis tomara como Template predeterminado uno llamado “default”. Los templates son utilizados por Canis para darle cuerpo al contenido. ¿No haz notado que todas las paginas web presentan un formado muy similar? Todas tienen un menu que se repite durante toda la pagina, un footer y un header, a veces hasta mucho más. En este sentido Canis permite manejar multiples templates por aplicacion, solo debemos especificarle que template utilizar en cada vista.
* **Styles**: Este parametro tambien es opcional, hay veces que una vista debe utilizar unos estilos en particular pero no deseamos colocarlos dentro de el archivo default.css ya que estos estilos solo seran utilizados una vez y no es necesario estar cargandolos en todas las paginas si no se van a utilizar. Para eso se creo esta propiedad, el nombre o URL del archivo especificado aqui Canis los buscará a partir del directorio css de nuestro proyecto.

Por utimo vamos a eplicar la siguiente linea:

**<consulta destino=’bienvenida’>getBienvenida</consulta>**

Al momento de visualizar una vista puede ser que necesitemos cargar contenido dinamico, por ejemplo, supongamos que vamos a cargar una vista donde queremos saludar al usuario por nombre y apellido segun su id de usuario. Bueno debemos definir un metodo que va a buscar el nombre de ese usuario en particular y la variable en donde lo guardará. En este caso hemos definido el metodo **getBienvenida** de la clase **InicioDelegate** que se encargara de guardar el nombre dentro de la variable **bienvenida.**

De esta manera cuando estemos en la vista solo deberemos pedir la variable bienvenida bajo el siguiente formato: $vars[“bienvenida”].