## URLs

La ruta de la url se establece en el controlador Base **es.jclm.cs.rarasclm.controller.BaseController**

Toma el atributo **route**

La clase **RouteParameters (es.jclm.cs.rarasclm.entities.RouteParameters)**

Tiene los siguientes prámetros:

/\*\* The modulo. \*/

**private** String modulo;

/\*\* The formulario. \*/

**private** String formulario;

/\*\* The accion. \*/

**private** String accion;

/\*\* The id. \*/

**private** String id;

/\*\* The base url. \*/

**private** String baseUrl;

Que indican el recurso ubicado en la url del suiente modo:

**{modulo}/{entidad}/{accion}/{ID|json}/{ID2?}**

Dónde:

El **módulo** indica el módulo de la aplicación:

{Inicio,Enfermedades,Casos,Pacientes,Admin}

La **entidad** indica la entidad con la que se va a trabajar

Por ejemplo: el módulo de enfermedades contiene las entidades: {enfRara,Cie9MC,Cie10,Snomed}

La **acción** es la operación que se va a realizar sobre la entidad {show,create,edit,delete}

**ID** es el identificador de entidad sobre el cual se realiza la operación.

**json** sobre el cuarto o quinto elemento lista entidades en formato json (cuando son menores de 1000). En caso contrario se necesita un argumento de búsqueda (variable search con request post) y muestra un error. Como caso particular el módulo de enfermedades se listan siempre todas.

Cuando json se escribe en el quinto parámetro se envía la entidad con el identificador **ID.**

## Estructura de la web (proceso de carga BaseModelView)

La clase **BaseModelView** Contiene la estructura de la aplicación y la instancia el contenedor spring

<bean id="baseModelView" class="es.jclm.cs.rarasclm.entities.BaseModelView"></bean>

WEB-INF/spring/spring-web.xml

El objeto se recoge inyectando la propiedad **base** del objeto BaseController cada vez que se hace una petición al servidor web.

**base** se usa , en la vista, para crear el menú principal de la aplicación y recoger parámetros en el contexto de aplicación.

MenuModel , List<MenuItem>

Cada Item de Menu tiene una referencia al MenuModulo y con su id se forma la url de acceso al recurso del módulo (uno ítem menu)

Cada Módulo tiene asociado un controlador. Cada itemMenu Tiene asociado un controlador, La notación del nombre de clase sigue la siguiente nomenclatura:

**{NombreModulo**+{**NombreEntidad**|*Vacio*}+”Controller”**}**

Cuando no se especifica el nombre de la entidad (es *Vacío*) indica que es el controlador que maneja la petición a la entrada principal del módulo.

**MenuModel** has many **MenuModulo** has many **MenuItem** (UML)

## Datos en memoria principal (caché) que se cargan en el inicio de la aplicación.

El contenedor de spring tiene un objeto DatosAuxiliaresCacheados para almacenar datos en memoria a los que se accede a menudo (como el caso de los datos de enfermedades o tablas auxiliares)

**DataContextRarasClmAppListener** implements ApplicationListener<ContextRefreshedEvent>

<bean id="rarasClmListener" class="es.jclm.cs.rarasclm.listeners.**DataContextRarasClmAppListener**"></bean>

<bean id="datosAuxiliares" class="es.jclm.cs.rarasclm.entities.**DatosAuxiliaresCacheados**"></bean>

WEB-INF/spring/spring-web.xml

IMPORTANTE – La capa de Servicio tiene que actualizar **DatosAuxiliaresCacheados[[1]](#footnote-1)** Cada vez que se actualice cualquier dato auxiliar en base de datos. (Poner ejemplo de flujo de datos o pila de llamadas de un cambio en estas tablas.

## Capa de servicio.

La usa los controladores de la capa web para cargar, actualizar y borrar modelos. Está implementada en el

## Tipos de asociación de una Enfermedad Rara.

* **Sin asociar** (No hay mapeo).
* **No Específica** (Detrás de una enfermedad con el código de mapeo puede haber una enfermedad rara pero no es una condición suficiente).
* **Específica** el código de la enfermedad es una enfermedad rara.

## EnfermedadRaraController.

Implementa el controlador que actualiza la entidad enfermedad rara. Es una codificación interna del registro de Castilla-La Mancha con mapeos a otros sistemas de codificación.

Las operaciones que implementa del modelo de negocio son:

* Crear una EnfermedadRara (de clm) nueva.
* Editar una EnfermedadRara
* Asociar (mapeo) a una enfermedad de la clasificación internacional 9ª Edición modificación clínica (CIE9MC).
* Asociar (mapeo) a una enfermedad de la clasificación internacional 10ª Edición (CIE10)
* Cambiar un tipo de asociación { Sin asociar, No Específica, Específica } ( por defecto sin asociar )

CHECK (Esto tiene que ser un conjunto de pruebas de integración?)

AccionRespuesta (Es lo que se devuelve al formulario para indicar el resultado la acción)

* **Actualización:**

enfermedades/enfrara/edit/[ID] **POST**

Controlador:

**public** String submitForm(

@ModelAttribute("enfermedadRaraModel") EnfermedadRaraView[[2]](#footnote-2) enf @ModelAttribute("resultado") AccionRespuesta resultado,

SessionStatus status)

Recibe la entidad EnfermedadRaraView

Llama a la capa de servicio update(enf)

Devuelve atributo resultado

1. Se tendría que implementar de algún otro modo. Seleccionando las entidades de persistencia en memoria de la App y después, de forma automática que se actualizarían. (Creación de un Aspecto sobre los servicios de estas entidades?) [↑](#footnote-ref-1)
2. EnfermedadRaraView es una clase de vista de EnfermedadRara con Listas en vez de Conjunto para establecer el orden de los elementos de colecciones en la vista (se tendría que implementar una adaptador o investigar que parte del frameworl MVC implementa algo parecido al patrón ModelView ( recuerda que estás trabajando con **java (it’s an ugly language**) ) [↑](#footnote-ref-2)