Universidad Nacional Autónoma de México

Dirección General de Cómputo de Tecnologías de Información y Comunicación

Diplomado

Desarrollo de Sistemas con Tecnología Java

Ejercicio Ocho

“Métodos de Fabrica”

Mtro. ISC. Miguel Ángel Sánchez Hernández

Tabla de contenido

1. Copiar un proyecto en en IntelliJ IDEA

Para copiar un proyecto, hacemos lo siguiente:

1. Señalamos el proyecto spring-core-autowire
2. Apretamos Ctrl + C
3. Luego Ctrl + V
4. New Name: spring-core-factory
5. En el archivo pom.xml cambiar las siguientes líneas
    - <artifactId>spring-core-autowire</artifactId> por lo siguiente
        - <artifactId>spring-core-factory</artifactId>
    - Cambiar las etiquetas <name> con
        - <name> spring-core-factory </name>
    - Cambiar la etiqueta <description> con
        - <description> Ejemplo de método de fábrica con Spring </description>
6. Actualizar maven

2. Paquetes

Crear los siguientes paquetes:

- dgtic.core.inicio
- dgtic.core.modelo
- dgtic.core.servicio

Borrar el contenido del paquete modelo.

3. Crear la clase Persona

La clase Persona debe de tener el siguiente código:

- Nombre de la clase: Persona
- Paquete: dgtic.core.modelo

```
package dgtic.core.modelo;
import dgtic.core.servicio.FabricaCoches;

public class Persona {
    private String nombre;
    private FabricaCoches coche;

    public Persona() {
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public FabricaCoches getCoche() {
        return coche;
    }
```

```
    public void setCoche(FabricaCoches coche) {
        this.coche = coche;
    }

    @Override
    public String toString() {
        return "Persona{" +
                "nombre='" + nombre + '\'' +
                '}';
    }
}
```

4. Crear un enum TiposCarro

Enum TiposCarro debe de tener el siguiente código:

- Nombre del enum: TiposCarro
- Paquete: dgtic.core.modelo

```
package dgtic.core.modelo;

public enum TiposCarro {
    DEPORTIVO,FAMILIAR,TODOTERRENO
}
```

5. Crear la interfaz ModeloCoche

Creamos la interfaz ModeloCoche con las siguientes características.

- Nombre de la interfaz: ModeloCoche
- Paquete: dgtic.core.servicio

```
package dgtic.core.servicio;
public interface ModeloCoche {
    public void crear();
}
```

6. Crear la clase Deportivo

Creamos la clase Deportivo con las siguientes características.

- Nombre de la clase: Deportivo
- Paquete: dgtic.core.servicio

```
package dgtic.core.servicio;
public class Deportivo implements ModeloCoche{

    @Override
    public void crear() {
        System.out.println("Crear carro deportivo");
    }
}
```

7. Crear la clase Familiar

Creamos la clase Familiar con las siguientes características.

- Nombre de la clase: Familiar
- Paquete: dgtic.core.servicio

```
package dgtic.core.servicio;
public class Familiar implements ModeloCoche{
    @Override
    public void crear() {
        System.out.println("Crear carro familiar");
    }
}
```

8. Crear la clase TodoTerreno

Creamos la clase Familiar con las siguientes características.

- Nombre de la clase: TodoTerreno
- Paquete: dgtic.core.servicio

```
package dgtic.core.servicio;
public class TodoTerreno implements ModeloCoche{
    @Override
    public void crear() {
        System.out.println("Crear carro todo terreno");
    }
}
```

9. Crear la clase FabricaCoches

Creamos la clase FabricaCoches con las siguientes características.

- Nombre de la clase: FabricaCoches
- Paquete: dgtic.core.servicio

```
package dgtic.core.servicio;
import dgtic.core.modelo.TiposCarro;
public class FabricaCoches {
    private static FabricaCoches fabricaCoches=new FabricaCoches();
    public static FabricaCoches getInstance(){
        return fabricaCoches;
    }
    public ModeloCoche getModeloCoche(TiposCarro tipo) throws IllegalAccessException {
        if(tipo.equals(TiposCarro.DEPORTIVO)){
            return new Deportivo();
        }else if(tipo.equals(TiposCarro.FAMILIAR)){
            return new Familiar();
        }else if(tipo.equals(TiposCarro.TODOTERRENO)){
            return new TodoTerreno();
        }
        throw new IllegalAccessException("No existe ese tipo de carro");
    }
}
```

10. Archivo de configuración bean-configuration.xml

Modificamos el archivo bean-configuration.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd">
   <!-- bean definitions here -->
   <bean id="persona" class="dgtic.core.modelo.Persona">
      <property name="nombre" value="Pedro"/>
       <property name="coche" ref="servicio"/>
   </bean>
</beans>
```

11. Archivo de configuración bean-services.xml

Crear el archivo bean-services.xml, en el paquete dgtic.core.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd">
   <!-- bean definitions here -->
   <bean id="servicio" class="dgtic.core.servicio.FabricaCoches"
         factory-method="getInstance">
   </bean>
</beans>
```

12. Clase Inicio

Modificamos la clase Inicio como sigue:

```
package dgtic.core.inicio;
import dgtic.core.modelo.Persona;
import dgtic.core.modelo.TiposCarro;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

public class Inicio {
    public static void main(String[] args) throws IllegalAccessException {
        ApplicationContext contexto = new ClassPathXmlApplicationContext(
                new String[] { "/src/main/java/dgtic/core/xml/bean-configuration.xml",
                        "/src/main/java/dgtic/core/xml/bean-services.xml"});
        Persona persona = (Persona) contexto.getBean("persona");
        persona.getCoche().getModeloCoche(TiposCarro.DEPORTIVO).crear();
        System.out.println(persona.toString());
        System.out.println("--------");
        Persona personaDos = (Persona) contexto.getBean("persona");
        personaDos.setNombre("Tomas");
        personaDos.getCoche().getModeloCoche(TiposCarro.FAMILIAR).crear();
        System.out.println(personaDos.toString());
        System.out.println("--------");
        System.out.println(persona.toString());
        System.out.println("--------");
        ((ClassPathXmlApplicationContext) contexto).close();

    }
}
```

Correar la aplicación.