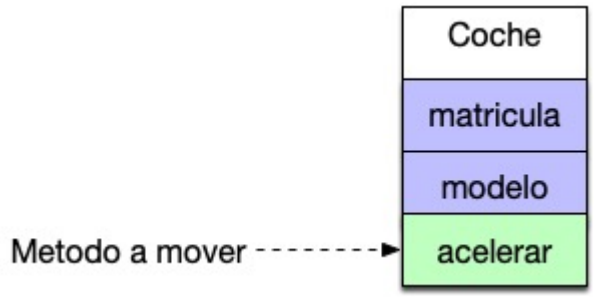
Usar Eclipse refactor move es muy habitual cuando tenemos que modelar un conjunto de clases . Ya que casi siempre sucede lo mismo , la responsabilidades que hemos asignado en un principio a una clase acabamos considerando que quizás debieran estar en otra y necesitamos mover un método de una clase a otra. Este tipo de operativa se puede realizar de una manera muy sencilla con la opción de Eclipse refactor move .Vamos a verlo para ello construimos la típica clase Coche:



Su código es el siguiente:

```
package com.arquitecturajava;

public class Coche {

	private String modelo;
	private String matricula;
	public String getModelo() {
		return modelo;
	}
	public void setModelo(String modelo) {
		this.modelo = modelo;
	}
```

```
	public String getMatricula() {
		return matricula;
	}
	public void setMatricula(String matricula) {
		this.matricula = matricula;
	}
	public void acelerar() {
		System.out.println("el coche acelera");
	}
}
```

Eclipse Refactor Move

Como podemos ver la clase dispone del método acelerar le invocamos en el programa main y el coche acelera.

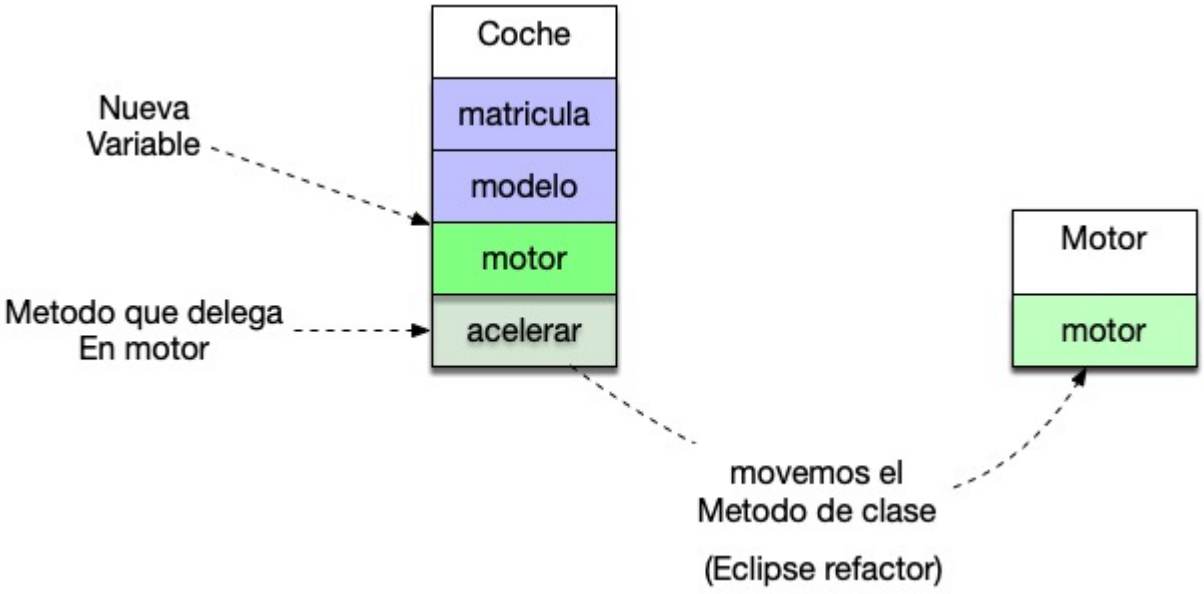
```
package com.arquitecturajava;

public class Principal {

	public static void main(String[] args) {
		Coche c= new Coche();
		c.acelerar();
	}

}
```

Esto de entrada parece que esta muy bien pero es probable que rápidamente aparezca la clase Motor y con esta clase queramos mover el método acelerar a ella y que la clase Coche lo ejecute por delegación .



¿Cómo podemos hacer esto de una forma automática? .Con Eclipse es realmente sencillo vamos a crear la clase Motor y modificar la clase Coche para que contenga la variable Motor.

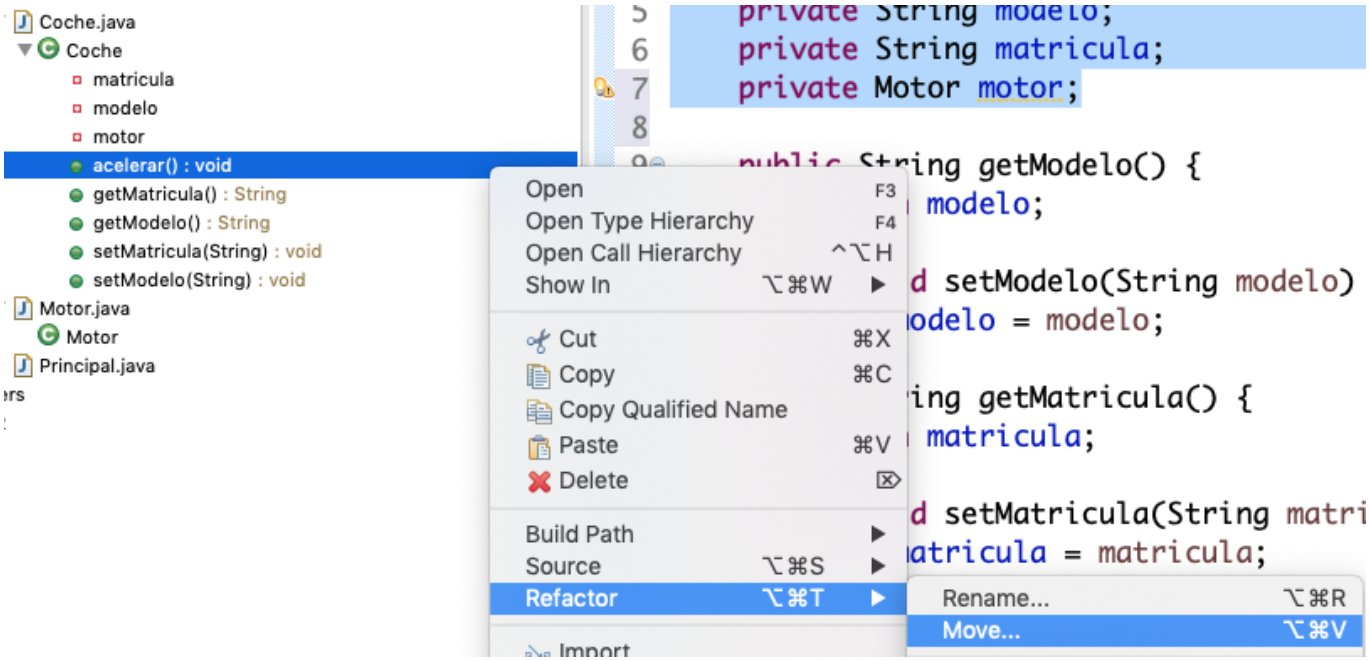
```
package com.arquitecturajava;

public class Motor {
}
```

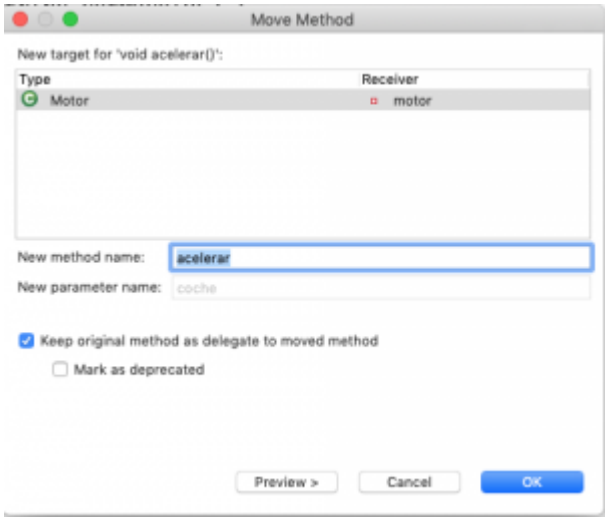
```
        public class Coche {

        private String modelo;
        private String matricula;
        private Motor motor;
.........resto de codigo....
```

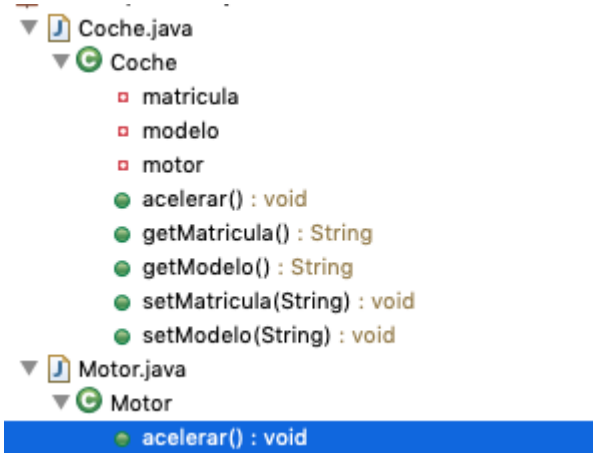
Una vez tenemos estas dos cosas ya podemos hacer uso del refactoring de Eclipse refactor move.



Una vez pulsemos sobre esta opción nos aparecerá una ventana con el movimiento que eclipse va a ejecutar:



Al tener un ejemplo tan sencillo ya nos indica a qué clase va a mover el método . Pulsamos aceptar y el método quedará movido:



Eso si el método acelerar de Coche no se borra sino que simplemente delega en el método de la clase Motor.Veamos el nuevo código:

```
package com.arquitecturajava;
```

```
public class Motor {

	public void acelerar() {
		System.out.println("el coche acelera");
	}
}
```

```
package com.arquitecturajava;

public class Coche {

	private String modelo;
	private String matricula;
	private Motor motor;
	public String getModelo() {
		return modelo;
	}
	public void setModelo(String modelo) {
		this.modelo = modelo;
	}
	public String getMatricula() {
		return matricula;
	}
	public void setMatricula(String matricula) {
		this.matricula = matricula;
	}
	public void acelerar() {
		motor.acelerar();
	}
```

```
}
```

Acabamos de ver como usar Eclipse refactor move y mover un método de una clase a otra de forma automática:

Otros artículos relacionados

1. Eclipse JPA y clases de dominio
2. Eclipse Git , Repositorios locales y remotos
3. Refactorings con Eclipse
4. Eclipse Pull up , Pull down y refactorings
5. https://www.eclipse.org/