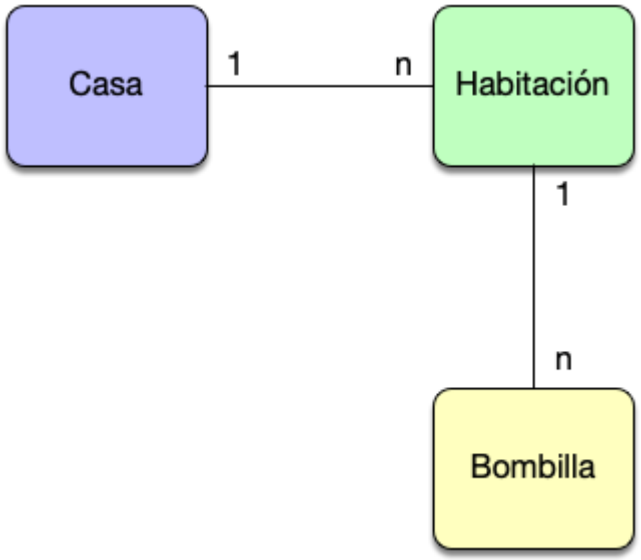
Hablar de Clases y Objetos Java y de como asignar sus responsabilidades es uno de los conceptos más habituales de programación orientada a objeto. Imaginemonos que disponemos de las siguientes tres clases relacionadas. Casa , Habitación y Bombilla. Una Habitación tiene varias bombillas y una Casa varias habitaciones si tenemos que construir un diagrama de clases nos encontraremos con un diagrama de composición de este estilo:



Por lo tanto una Casa contiene una ArrayList de Habitaciones y una Habitación un ArrayList de Bombillas. Hasta aquí todo es bastante natural , supongamos ahora que queremos saber cual es la bombilla que más watios gasta de la casa . Vamos a ver la relación de clases a nivel del código.

```
package com.arquitecturajava.ejemplo1;

import java.util.ArrayList;

public class Casa {
```

```
  private ArrayList<Habitacion> habitaciones=new
ArrayList<Habitacion>();

  public ArrayList<Habitacion> getHabitaciones() {
    return habitaciones;
  }

  public void setHabitaciones(ArrayList<Habitacion> habitaciones) {
    this.habitaciones = habitaciones;
  }

  public void addHabitacion(Habitacion h) {
    habitaciones.add(h);
  }
  public Bombilla bombillaMayorGasto() {
    Bombilla bombillaGasto= new Bombilla(0);
    for(Habitacion h :habitaciones) {
      for(Bombilla b: h.getBombillas()) {
        if (bombillaGasto.getPotencia()<b.getPotencia()) {
          bombillaGasto=b;
        }
      }
    }
    return bombillaGasto;
  }
}
```

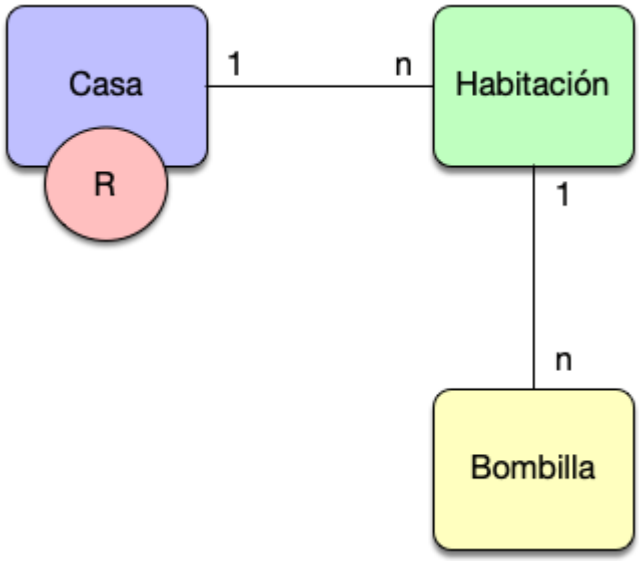
```
package com.arquitecturajava.ejemplo1;

import java.util.ArrayList;

public class Habitacion {
```

```
  private ArrayList<Bombilla> bombillas=new ArrayList<Bombilla>();

  public ArrayList<Bombilla> getBombillas() {
    return bombillas;
  }

  public void setBombillas(ArrayList<Bombilla> bombillas) {
    this.bombillas = bombillas;
  }
  public void addBombilla(Bombilla b) {
    bombillas.add(b);
  }

}
```

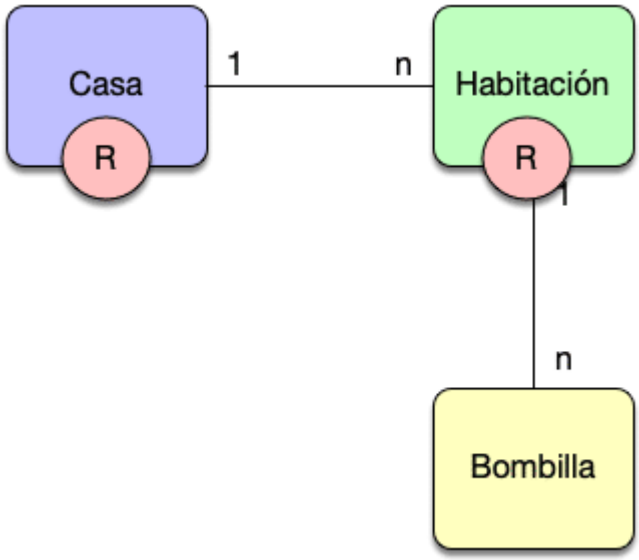
```
package com.arquitecturajava.ejemplo1;

public class Bombilla {

  private int potencia;

  public int getPotencia() {
    return potencia;
  }

  public void setPotencia(int potencia) {
    this.potencia = potencia;
  }

  public Bombilla(int potencia) {
    super();
    this.potencia = potencia;
```

```
  }
}
```

En estos momentos es la clase Casa la que se encarga de calcular cuál es la bombilla que gasta más de toda la Casa . Si revisamos a nivel conceptual la asignación de responsabilidades nos puede dar la sensación de que es correcta. Al final es la casa la encargada de saber cuál es la bombilla que gasta más.



Ahora bien en el mundo del diseño orientado a objeto lo más habitual es que dividamos las responsabilidades y que cada clase se encargue de aquellas que le son mas afines. Una mejor solución sería que la clase Habitación se encargará de saber cual es la bombilla que más consume dentro de ella:



Veamos la nueva versión del código a nivel de clases:

```
package com.arquitecturajava.ejemplo2;

import java.util.ArrayList;

public class Casa {

  private ArrayList<Habitacion> habitaciones=new
ArrayList<Habitacion>();

  public ArrayList<Habitacion> getHabitaciones() {
    return habitaciones;
  }

  public void setHabitaciones(ArrayList<Habitacion> habitaciones) {
    this.habitaciones = habitaciones;
  }
```

```
  public void addHabitacion(Habitacion h) {
    habitaciones.add(h);
  }
  public Bombilla bombillaMayorGasto() {
    Bombilla bombillaGasto= new Bombilla(0);
    for(Habitacion h :habitaciones) {
      if
(bombillaGasto.getPotencia()<h.bombillaMayorGasto().getPotencia()) {
        bombillaGasto=h.bombillaMayorGasto();
      }
    }
    return bombillaGasto;
  }
}
```

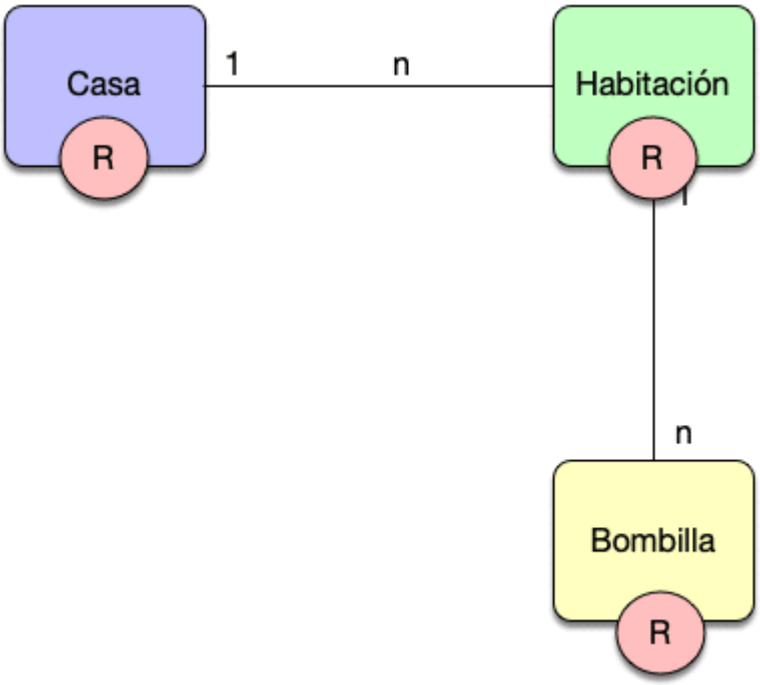
```
package com.arquitecturajava.ejemplo2;

import java.util.ArrayList;

public class Habitacion {

  private ArrayList<Bombilla> bombillas=new ArrayList<Bombilla>();

  public ArrayList<Bombilla> getBombillas() {
    return bombillas;
  }

  public void setBombillas(ArrayList<Bombilla> bombillas) {
    this.bombillas = bombillas;
  }
  public void addBombilla(Bombilla b) {
    bombillas.add(b);
```

```
  }

  public Bombilla bombillaMayorGasto() {
    Bombilla bombillaGasto= new Bombilla(0);
    for(Bombilla b: bombillas) {
      if (bombillaGasto.getPotencia()<b.getPotencia()) {
        bombillaGasto=b;
      }
    }
    return bombillaGasto;
  }
}
```

```
package com.arquitecturajava.ejemplo2;

public class Bombilla {

  private int potencia;

  public int getPotencia() {
    return potencia;
  }

  public void setPotencia(int potencia) {
    this.potencia = potencia;
  }

  public Bombilla(int potencia) {
    super();
    this.potencia = potencia;
  }
}
```

Clases y Objetos Java con sus responsabilidades

Ahora el código parece más correcto ya que las responsabilidades están mas divididas. Aun así podemos afinarlo un poco mas y que la responsabilidad de si una bombilla gasta más que otra recaiga en la clase bombilla.



Vamos a ver como queda el código fuente:

```
package com.arquitecturajava.ejemplo3;

import java.util.ArrayList;

public class Casa {

  private ArrayList<Habitacion> habitaciones=new
ArrayList<Habitacion>();
```

```
  public ArrayList<Habitacion> getHabitaciones() {
    return habitaciones;
  }

  public void setHabitaciones(ArrayList<Habitacion> habitaciones) {
    this.habitaciones = habitaciones;
  }

  public void addHabitacion(Habitacion h) {
    habitaciones.add(h);
  }
  public Bombilla bombillaMayorGasto() {
    Bombilla bombillaGasto= new Bombilla(0);
    for(Habitacion h :habitaciones) {
      bombillaGasto=
bombillaGasto.mayorPotencia(h.bombillaMayorGasto());
    }
    return bombillaGasto;
  }
}
```

```
package com.arquitecturajava.ejemplo3;

import java.util.ArrayList;

public class Habitacion {

  private ArrayList<Bombilla> bombillas=new ArrayList<Bombilla>();

  public ArrayList<Bombilla> getBombillas() {
    return bombillas;
  }
```

```
  public void setBombillas(ArrayList<Bombilla> bombillas) {
    this.bombillas = bombillas;
  }
  public void addBombilla(Bombilla b) {
    bombillas.add(b);
  }

  public Bombilla bombillaMayorGasto() {
    Bombilla bombillaGasto= new Bombilla(0);
    for(Bombilla b: bombillas) {
      bombillaGasto= bombillaGasto.mayorPotencia(b);
    }
    return bombillaGasto;
  }
}
```

```
package com.arquitecturajava.ejemplo3;

public class Bombilla {

  private int potencia;

  public int getPotencia() {
    return potencia;
  }

  public void setPotencia(int potencia) {
    this.potencia = potencia;
  }

  public Bombilla(int potencia) {
    super();
```

```
    this.potencia = potencia;
  }
  public Bombilla mayorPotencia(Bombilla otra) {
    if (this.getPotencia()>otra.getPotencia()) {
      return this;
    }else {
      return otra;
    }
  }
}
```

De esta manera hemos dividido de forma más homogénea las responsabilidades en cuando a Clases y Objetos Java. Permitiendo una mayor reutilización del código a futuro.

1. Eclipse JPA y clases de dominio
2. Java Herencia vs Interfaces
3. Interface Segregation Principle y Spring Data
4. Diseño orientado a objeto