```
class Employee
{
     int id,String name, Addres addr
     default & parametrized Constructors
     getters & setters
     toString()
}
```

```
class Address
{
     int doorno, String street,String city
     default & parametrized Constructors
     getters & setters
     toString()
}
```

Employee has a Address
Employee has a id
Employee has a name

Dependent : Employee
Dependencies : primitive : Id,Name
                    Reference  : addr

```
Employee e=new Employee()
e.setId(11);
e.setName("aaa");
e.setAddr(new Address(12,"a street","city1"))
```

```
Employee e=new Employee()
e.setId(11);
e.setName("aaa");
Address a=new Address(12,"a street","city1")
e.setAddr(a)
```

```
Employee e=new Employee()
e.setId(11);
e.setName("aaa");
        Address a=new Address()
        a.setDoorNo(101)
        a.setStreet("a street")
        a.setCity("City1")
e.setAddr(a)
```

```
Employee e=new Employee(11,"Anand",new Address(12,"a street","city1"))
```

**Manual** wiring will difficult in large App -----------------> AutoWiring

Explicit

In Explicit Autowiring Configurations are done using XML files

Implicit

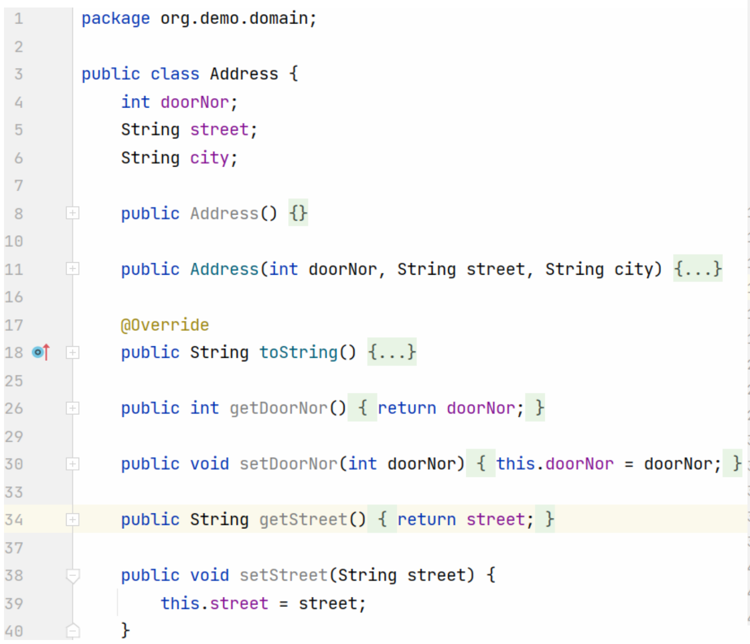
In Implicit Autowiring Configurations are done using Annotations- @Autowired

1. property
2. Setter
3. Constructor

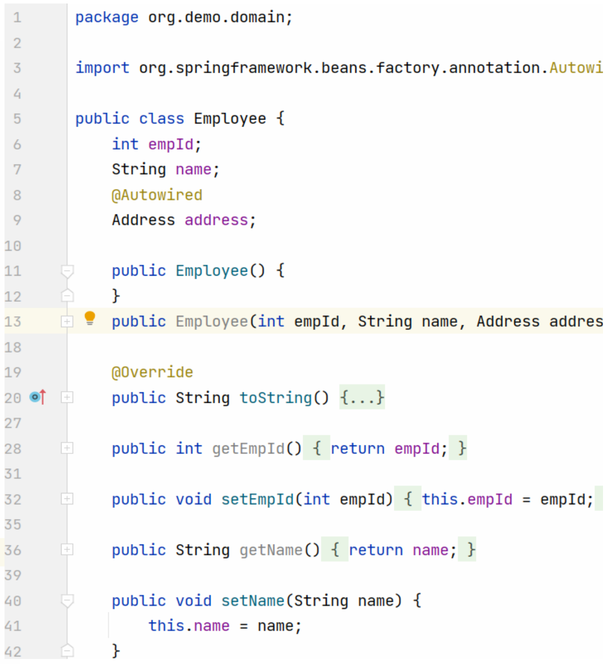
**Property Autowiring**

Step1 Add Spring context dependency in pom.xml

step2 Create Domain classes Address & Employee and add @Autowired on top of property

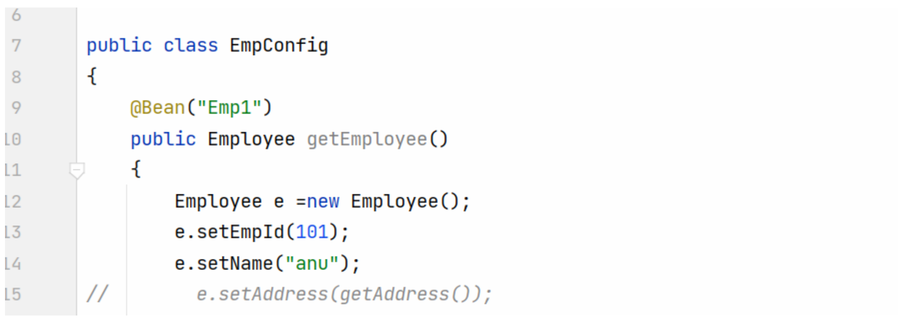


```
package org.demo.domain;

public class Address {
    int doorNor;
    String street;
    String city;

    public Address() {}

    public Address(int doorNor, String street, String city) {...}

    @Override
    public String toString() {...}

    public int getDoorNor() { return doorNor; }

    public void setDoorNor(int doorNor) { this.doorNor = doorNor; }

    public String getStreet() { return street; }

    public void setStreet(String street) {
        this.street = street;
    }
```

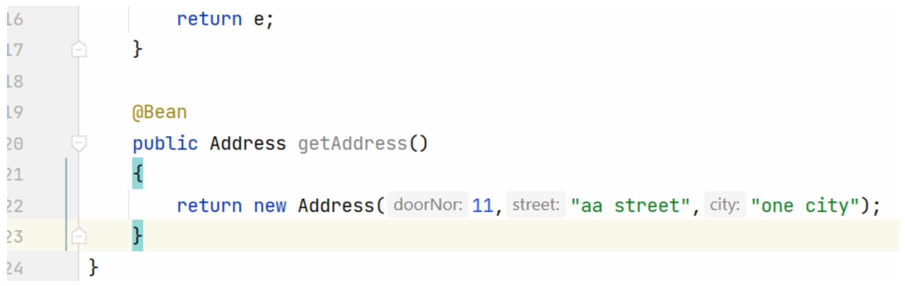


```
package org.demo.domain;

import org.springframework.beans.factory.annotation.Autowi

public class Employee {
    int empId;
    String name;
    @Autowired
    Address address;

    public Employee() {
    }
    public Employee(int empId, String name, Address addres

    @Override
    public String toString() {...}

    public int getEmpId() { return empId; }

    public void setEmpId(int empId) { this.empId = empId;

    public String getName() { return name; }

    public void setName(String name) {
        this.name = name;
    }
```

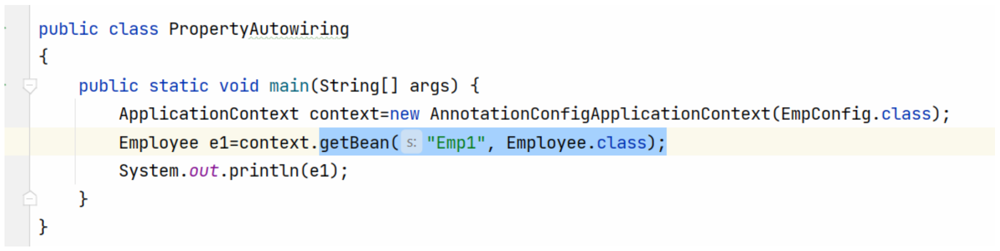
Step 3 create Config class



```
public class EmpConfig
{
    @Bean("Emp1")
    public Employee getEmployee()
    {
        Employee e =new Employee();
        e.setEmpId(101);
        e.setName("anu");
//        e.setAddress(getAddress());
```


```
        return e;
    }

    @Bean
    public Address getAddress()
    {
        return new Address( doorNor: 11, street: "aa street", city: "one city");
    }
}
```

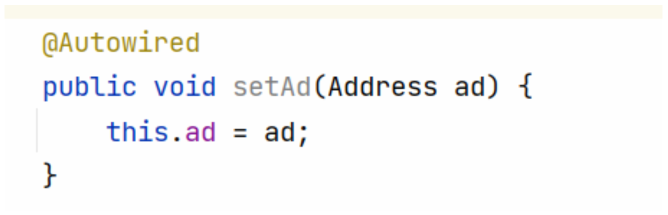
Step 4     Creating     main class


```
public class PropertyAutowiring
{
    public static void main(String[] args) {
        ApplicationContext context=new AnnotationConfigApplicationContext(EmpConfig.class);
        Employee e1=context.getBean( s: "Emp1", Employee.class);
        System.out.println(e1);
    }
}
```

**SETTER Autowiring**

Step1     Add Spring context dependency in pom.xml

step2     Create Domain classes Address & Employee and add @Autowired on top of setter method


```
@Autowired
public void setAd(Address ad) {
    this.ad = ad;
}
```

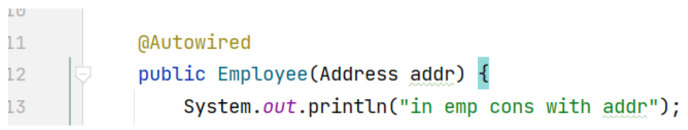
Step 3     create Config class

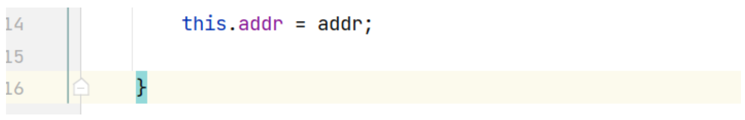
step 4     Create main

**Constructor Autowiring**

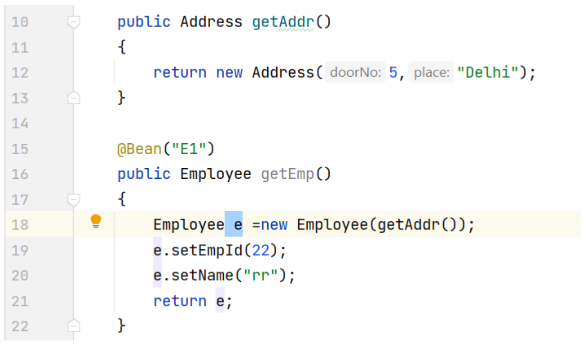
Step1     Add Spring context dependency in pom.xml

step2     Create Domain classes Address & Employee and add @Autowired on top of constructor


```
    @Autowired
    public Employee(Address addr) {
        System.out.println("in emp cons with addr");
```


```
        this.addr = addr;

    }
```

Step 3 create Config class


```
public Address getAddr()
{
    return new Address( doorNo: 5, place: "Delhi");
}

@Bean("E1")
public Employee getEmp()
{
    Employee e =new Employee(getAddr());
    e.setEmpId(22);
    e.setName("rr");
    return e;
}
```

step 4 Create main

```
ired;



ss) {...}


}
```