Contents

1 Documentación da Clase `Contacto`

1.1 Que é esta clase?

Imaxina que esta clase é coma unha **ficha de contacto** na túa axenda telefónica. Cada ficha contén: - A persoa (nome, apelidos, NIF) - O seu teléfono - O seu email
- Unha descrición (ex: “Amigo”, “Compañeiro”) - Un número de identificación único

A clase `Contacto` organiza toda esta información nun só obxecto, permitíndote crear, modificar e buscar contactos de forma ordenada.

1.2 Partes Importantes

1.2.1 Atributos Principais

```
private int numero;           // Número único (comeza en -1)
private final String descripcion; // Descrición (non cambia)
```

```
private String direccion;     // Enderezo (opcional)
private String telefono;      // Teléfono (validado)
private String email;         // Email (validado)
private final Persoa persoa;  // Persoa asociada (non cambia)
```

Chave para principiantes: - `final` = valor que non pode cambiar unha vez establecido - `private` = só se pode acceder mediante métodos específicos - `numero = -1` = indica que o contacto aínda non foi gardado

---

1.2.2 Como Crear un Contacto

```
Contacto c = new Contacto(persoa, "612345678", "email@email.com", "Amigo");
```

1.2.2.1 Constructor Completo (Recomendado)

```
// Sen descrición
Contacto c1 = new Contacto(persoa, "612345678", "email@email.com");

// Só con teléfono
Contacto c2 = new Contacto(persoa, "612345678");
```

1.2.2.2 Constructores Simplificados **Importante:** O teléfono debe ter 9 díxitos e o email formato correcto. Se non, o programa fallará.

1.2.3 Métodos Útiles

```
contacto.setNumero(5);            // Só para a axenda
contacto.setDireccion("Rúa 123"); // Opcional
contacto.setTelefono("698765432"); // Valida automaticamente
contacto.setEmail("novo@email.com"); // Valida automaticamente
```

1.2.3.1 Para Modificar Datos

```
Persoa p = contacto.getPersoa();
String tel = contacto.getPhone();
String desc = contacto.getDescripcion();
int num = contacto.getNumero();
String em = contacto.getEmail();
```

1.2.3.2 Para Obter Datos

```java
System.out.println(contacto);
// Saída: "5)Ana García: 612345678 (ana@email.com) - Colega"
```

1.2.3.3 Para Mostrar o Contacto

1.3 Exemplo Práctico

```java
import axenda.model.Persoa;
import axenda.model.Contacto;

public class Exemplo {
    public static void main(String[] args) {
        try {
            // Crear persoa e contacto
            Persoa ana = new Persoa("12345678Z", "Ana", "García");
            Contacto contacto = new Contacto(ana, "612345678",
                                             "ana@email.com", "Colega");

            // Asignar número (feito pola axenda)
            contacto.setNumero(1);

            // Mostrar
            System.out.println(contacto);
            // Saída: "1)Ana García: 612345678 (ana@email.com) - Colega"

            // Modificar teléfono (valida automaticamente)
            contacto.setTelefono("611223344");

        } catch (IllegalArgumentException e) {
            System.err.println("Erro: " + e.getMessage());
        }
    }
}
```

1.4 Validación Automática

1.4.1 No Constructor e Modificacións

Teléfono: Debe ter exactamente 9 díxitos
Email: Debe ter formato correcto (con @)

Exemplos que falan:

```java
// Isto falla (teléfono incorrecto)
new Contacto(persoa, "61234567", "email@email.com", "Desc");

// Isto falla (email incorrecto)
contacto.setEmail("email.sen.arroba.com");
```

Solución:

```java
try {
    contacto.setTelefono("612345678"); // Valida automaticamente
} catch (IllegalArgumentException e) {
    System.out.println("Teléfono incorrecto");
}
```

1.5 Boas Prácticas para Principiantes

1.5.1 1. Sempre Usa try-catch

```java
try {
    Contacto novo = new Contacto(p, tel, email, desc);
} catch (IllegalArgumentException e) {
    System.out.println("Erro: " + e.getMessage());
}
```

1.5.2 2. Non Modifiques Atributos Directamente

```java
// MAL! Non fagas isto:
contacto.telefono = "612345678";

// BEN! Fai sempre isto:
contacto.setTelefono("612345678");
```

1.5.3 3. Usa os Constructores Correctos

- Tes todos os datos? Usa o constructor completo
- Non tes email? Usa o constructor con só teléfono
- Non tes descrición? Usa o constructor sen descrición

1.5.4 4. Lembra o que Non Pode Cambiar

- A **persoa asociada** non se pode cambiar
- A **descrición** non se pode cambiar
- O **número** só o debe asignar a axenda

1.5.5 5. Comproba o Número

```
if (contacto.getNumero() == -1) {
    System.out.println("Contacto non gardado");
} else {
    System.out.println("Gardado na posición: " + contacto.getNumero());
}
```

1.6 Erros Comúns e Solucións

1.6.1 Erro 1: Datos Incorrectos

```
// Isto falla
contacto.setTelefono("teléfono");
```

Solución: Deixa que a clase valide automaticamente:

```
try {
    contacto.setTelefono("612345678"); // Valida antes de gardar
} catch (IllegalArgumentException e) {
    System.out.println("Formato incorrecto");
}
```

1.6.2 Erro 2: Persoa Null

```
// Isto falla
Contacto erro = new Contacto(null, "612345678", "email", "Desc");
```

Solución: Crea sempre a persoa primeiro:

```
Persoa p = new Persoa("12345678Z", "Nome", "Apelidos");
Contacto c = new Contacto(p, "612345678", "email", "Desc");
```

1.6.3 Erro 3: Número Incorrecto

```
// MAL! Non asignes números manualmente
contacto.setNumero(100);
```

Solución: Deixa que a axenda o faga:

```
axenda.guardarContacto(contacto); // A axenda asigna o número
```

1.7 Por que é Importante?

Esta clase ensíñache conceptos fundamentais:

Encapsulación: Os datos están protexidos
Validación: Comproba datos antes de crear obxectos
Reutilización: Crea moitos contactos co mesmo patrón
Manexo de erros: Usa excepcións para problemas de formato
Boas prácticas: Separa creación de obxectos da lóxica de negocio

---

1.8 Resumo Rápido

Concepto	Explicación
Constructor	Crea un novo contacto con validación
Validación	Comproba teléfono e email automáticamente
setNumero()	Só para uso interno da axenda
getPersoa()	Obtén a persoa asociada
toString()	Mostra o contacto dun xeito legible

1.9 Seguintes Pasos

1. **Practica**: Crea un programa que engada 3 contactos e amosalos
2. **Proba erros**: Intenta crear contactos con datos incorrectos
3. **Combina**: Usa esta clase coa clase `ContactosDAO`
4. **Amplía**: Engade busca por nome ou teléfono

Esta clase é un excelente comezo para entender como organizar datos nun programa. Cando a domines, estarás preparado para aplicacións máis complexas!