Contents

1 Documentación da Clase `Persoa`

1.1 Que é esta clase?

Imaxina que esta clase é coma un **DNI dixital** que contén a información básica dunha persoa: - O seu NIF (Número de Identificación Fiscal) - O seu nome - Os seus apelidos

A clase `Persoa` garda esta información de forma segura e valida que o NIF teña formato correcto antes de crear o obxecto.

1.2 Partes Importantes

1.2.1 Atributos (Datos que Garda)

```
private final String nif;       // NIF da persoa (ex: "12345678Z")
private final String nome;      // Nome da persoa (ex: "Ana")
private final String apelidos;  // Apelidos da persoa (ex: "García")
```

Chave para principiantes: - `final` = Estes valores non poden cambiar unha vez creados - `private` = Só se poden ler mediante métodos especiais - O NIF é a **identidade única** da persoa

---

1.2.2 Como Crear unha Persoa

```
// Forma correcta (con NIF válido)
Persoa ana = new Persoa("12345678Z", "Ana", "García");

// Forma incorrecta (NIF inválido - fallará)
Persoa erro = new Persoa("1234567", "Ana", "García");
```

Importante: O NIF debe ter 8 números + 1 letra correcta. Se non, o programa dará un erro.

1.2.3 Métodos Útiles

```
String nif = ana.getNif();       // "12345678Z"
String nome = ana.getNome();     // "Ana"
String apel = ana.getApelidos(); // "García"
```

1.2.3.1 Para Obter Datos

```
System.out.println(ana);
// Saída: "[12345678Z] García, Ana"
```

1.2.3.2 Para Mostrar a Persoa

1.3 Exemplo Práctico

```
import axenda.model.Persoa;

public class ExemploPersoa {
    public static void main(String[] args) {
        try {
            // Crear unha persoa válida
            Persoa carlos = new Persoa(
                "87654321X",
                "Carlos",
                "Rodríguez"
```

```
            );

            // Mostrar a persoa
            System.out.println("Persoa creada:");
            System.out.println(carlos);
            // Saída: "[87654321X] Rodríguez, Carlos"

            // Obter datos individuais
            System.out.println("NIF: " + carlos.getNif());
            System.out.println("Nome: " + carlos.getNome());

        } catch (IllegalArgumentException e) {
            System.err.println("Erro: " + e.getMessage());
            System.err.println("Lembra: O NIF debe ter 8 números + 1 letra");
        }
    }
}
```

1.4 Validación do NIF

1.4.1 Que se valida?

Lonxitude: Exactamente 9 caracteres
Formato: 8 números + 1 letra
Letra de control: A letra debe corresponder cos números

1.4.2 Exemplos Válidos vs Inválidos

```
// VÁLIDOS
new Persoa("12345678Z", "Ana", "García");
new Persoa("87654321X", "Carlos", "Rodríguez");

// INVÁLIDOS (dan erro)
new Persoa("1234567Z", "Ana", "García");     // Só 7 números
new Persoa("123456789", "Ana", "García");    // 9 números, non 8+1
new Persoa("12345678A", "Ana", "García");    // Letra incorrecta
```

Solución: Usa sempre NIFs válidos. Se non estás seguro, comproba antes:

```
try {
    Input.testNif("12345678Z"); // Valida antes de crear
    Persoa p = new Persoa("12345678Z", "Ana", "García");
} catch (IllegalArgumentException e) {
    System.out.println("NIF incorrecto");
}
```

1.5 Boas Prácticas para Principiantes

1.5.1 1. Sempre Usa try-catch

```
try {
    Persoa nova = new Persoa(nif, nome, apelidos);
} catch (IllegalArgumentException e) {
    System.out.println("Erro no NIF: " + e.getMessage());
}
```

1.5.2 2. Non Modifiques Atributos Directamente

```
// MAL! Non fagas isto:
ana.nif = "11111111H";

// BEN! Fai sempre isto:
// (Non hai métodos set porque é inmutable)
```

1.5.3 3. Usa os Métodos Get Correctamente

```
// BEN
String n = ana.getNif();
String no = ana.getNome();

// MAL (confusión común)
String n = ana.nif; // Isto non funciona!
```

1.5.4 4. Lembra que é Inmutable

Unha vez creada unha persoa, os seus datos **non poden cambiar**:

```
// Isto NON é posible:
ana.setNome("María"); // Non existe este método!
ana.nif = "11111111H"; // Non se pode modificar!
```

1.6 Erros Comúns e Solucións

1.6.1 Erro 1: NIF Incorrecto

```java
// Isto falla
Persoa p = new Persoa("1234567", "Ana", "García");
```

Solución: Comproba o formato do NIF: - 8 números + 1 letra - A letra debe ser correcta para eses números - Exemplo correcto: "12345678Z"

1.6.2 Erro 2: Esquecer try-catch

```java
// Isto pode pechar o programa se o NIF é incorrecto
Persoa p = new Persoa(nifUsuario, nome, apelidos);
```

Solución: Envolve sempre en try-catch:

```java
try {
    Persoa p = new Persoa(nif, nome, apelidos);
} catch (IllegalArgumentException e) {
    System.out.println("Por favor, introduce un NIF válido");
}
```

1.6.3 Erro 3: Intentar Modificar Datos

```java
// Isto non funciona!
ana.setNome("María");
ana.setNif("11111111H");
```

Solución: Se necesitas cambiar datos, crea unha nova persoa:

```java
// Para cambiar o nome, crea un novo obxecto
Persoa anaNova = new Persoa(ana.getNif(), "María", ana.getApelidos());
```

1.7 Por que é Importante?

Esta clase ensíñache conceptos fundamentais:

Validación de datos: Comproba que a información sexa correcta
Inmutabilidade: Algúns datos non deben cambiar unha vez creados
Encapsulamento: Os datos están protexidos e só se accede mediante métodos
Identidade única: Cada persoa ten un identificador único (o NIF)
Manexo de erros: Usa excepcións para indicar problemas de formato

---

1.8 Resumo Rápido

Concepto	Explicación
NIF	Identificador único (8 números + 1 letra)
final	Os datos non poden cambiar despois de crear
getNif()	Obtén o NIF da persoa
toString()	Mostra a persoa como "[NIF] Apelidos, Nome"
try-catch	Sempre usa para xestionar erros de NIF

1.9 Seguintes Pasos

1. **Practica**: Crea un programa que pida ao usuario NIF, nome e apelidos
2. **Proba erros**: Intenta crear persoas con NIFs incorrectos
3. **Combina**: Usa esta clase xunto coa clase `Contacto` para crear unha axenda
4. **Amplía**: Fai un programa que amose varias persoas nunha lista

Esta clase é un excelente comezo para entender como traballar con datos persoais de forma segura e correcta. Cando a domines, estarás preparado para construír aplicacións máis complexas!