En Java, cada componente tiene su propia convención de nomenclatura, que generalmente sigue las reglas de camel case de una forma u otra. Aquí están las convenciones para los componentes más comunes:

- **Paquetes (Packages)**: Los nombres de paquetes siempre deben estar en minúsculas. Si un nombre consta de varias palabras, estas se deben concatenar (sin espacios ni guiones), aunque en la práctica es común ver nombres de paquetes que son solo una palabra. Ejemplo: `com.micompania.proyecto`.
- **Clases e Interfaces**: Para las clases e interfaces, se utiliza UpperCamelCase (también conocido como PascalCase). Cada palabra comienza con una letra mayúscula, incluida la primera. Ejemplo: `MiClase`, `FiguraGeometrica`.
- **Métodos**: Los nombres de métodos siguen la convención lowerCamelCase, donde la primera letra de la primera palabra es minúscula, y la primera letra de cada palabra subsiguiente es mayúscula.
  Ejemplo: `calcularDistancia`, `enviarMensaje`.
- **Variables**: Al igual que los métodos, las variables también utilizan lowerCamelCase. Esto incluye instancias de objetos, variables locales, y parámetros. Ejemplo: `miVariableLocal`, `valorTotal`.
- **Constantes**: Las constantes en Java se escriben completamente en mayúsculas, con palabras separadas por guiones bajos (_).
  Ejemplo: `VALOR_MAXIMO`, `NUMERO_DE_EMPLEADOS`.
- **Tipos genéricos (Type Parameters)**: Por convención, los nombres de los parámetros de tipo genérico suelen ser letras mayúsculas simples. Las más comunes son `T` para "Type", `E` para "Element", `K` para "Key", `V` para "Value", y `R` para "Return". Ejemplo: `public class Caja<T> { ... }`.

Aquí tienes un ejemplo que muestra todas estas convenciones en uso:

JAVA

```java
package com.micompania.proyecto.utilidades; // Paquete en minúsculas

public class Calculadora { // Clase en UpperCamelCase

    public static final double PI = 3.14159; // Constante en mayúsculas con guiones bajos

    private double resultadoAcumulado; // Variable en lowerCamelCase

    public Calculadora() { // Constructor en UpperCamelCase
        this.resultadoAcumulado = 0.0;
    }

    public double sumar(double sumando) { // Método en lowerCamelCase
```

```
        resultadoAcumulado += sumando;
        return resultadoAcumulado;
    }

    public static <T extends Number> T maximo(T a, T b) {
// Método genérico con parámetro de tipo en mayúscula
        return (a.doubleValue() > b.doubleValue()) ? a :
b;
    }
}
```

Siguiendo estas convenciones se mejora la legibilidad del código y se facilita el trabajo en equipo, ya que otros desarrolladores pueden entender más fácilmente la estructura y el propósito de las diferentes partes del código.