Java Enum:

In Java, an enum, short for "enumeration," is a special data type that represents a fixed set of constants or values. Enumerations are often used to define a set of related constants, making your code more readable and less error-prone. Here are two code examples to help you understand Java enums.

Example 1: Basic Enum

In this example, we'll create a simple enum called `DayOfWeek` to represent the days of the week.

```java
public enum DayOfWeek {
    SUNDAY,
    MONDAY,
    TUESDAY,
    WEDNESDAY,
    THURSDAY,
    FRIDAY,
    SATURDAY
}
```

Now, let's see how you can use this enum in your Java code:

```java
public class EnumExample {
    public static void main(String[] args) {
        DayOfWeek today = DayOfWeek.FRIDAY;

        switch (today) {
            case MONDAY:
                System.out.println("It's Monday.");
                break;
            case FRIDAY:
                System.out.println("It's Friday.");
                break;
            default:
                System.out.println("It's some other day.");
        }
    }
}
```

In this code:

- We define an enum `DayOfWeek` with seven constants.
- In the `main` method, we create an instance of `DayOfWeek` called `today` and set it to `DayOfWeek.FRIDAY`.
- We use a `switch` statement to check the value of `today` and print a corresponding message.

Example 2: Enum with Properties

In this example, we'll create an enum called `Planet` that represents some planets in our solar system and assigns properties like their mass and radius to each planet.

```java
public enum Planet {
    MERCURY(3.303e+23, 2.4397e6),
    VENUS(4.869e+24, 6.0518e6),
    EARTH(5.972e+24, 6.371e6),
    MARS(6.419e+23, 3.3895e6),
    JUPITER(1.899e+27, 6.9911e7),
    SATURN(5.685e+26, 5.8232e7),
    URANUS(8.683e+25, 2.5362e7),
    NEPTUNE(1.024e+26, 2.4622e7);

    private final double mass;   // in kilograms
    private final double radius; // in meters

    Planet(double mass, double radius) {
        this.mass = mass;
        this.radius = radius;
    }

    public double getMass() {
        return mass;
    }

    public double getRadius() {
        return radius;
    }
}
```

In this code:

- We define an enum `Planet` with eight constants, each representing a planet in the solar system.
- Each planet has two properties: `mass` and `radius`, which are specific to that planet.
- We define a constructor that takes these properties as arguments and assigns them to the corresponding fields.
- We also provide getter methods to access the properties of each planet.

Now, you can use this enum to calculate properties of planets:

```java
public class EnumWithPropertiesExample {
    public static void main(String[] args) {
        Planet earth = Planet.EARTH;
        System.out.println("Earth's mass: " + earth.getMass() + " kg");
        System.out.println("Earth's radius: " + earth.getRadius() + " meters");
    }
}
```

In this code, we create an instance of `Planet` representing Earth and use its getter methods to access its properties.

These examples demonstrate how enums can be used to define a set of constants and even associate properties with each constant, making your code more organized and self-explanatory.