Here are detailed notes extracted and structured from the uploaded document:

ENUM in Java

Definition

- `enum` is used to define a group of named constants.
- Enables creation of enumerated data types, which can be user-defined.

Example

```
enum Month {
    JAN, FEB, MAR, ... DEC; // The semicolon is optional.
}
```

Internal Implementation

- Internally, enums are implemented using the class concept.
- Characteristics of enum constants:
  - Implicitly `public static final`.
  - Internally static, accessible via `enum` name.
  - Each constant is a reference variable to its `enum` type object.
- `toString()` method:
  - Internally implemented to return the name of the constant.

---

enum vs Enum vs Enumeration

Aspect	`enum` (keyword)	`Enum` (class)	`Enumeration` (interface)
Definition	A keyword to define constants.	A class in `java.lang`. Acts as a base class for all enums.	An interface in `java.util`. Retrieves objects one by one from collections.
Inheritance	Directly extends `Enum` class.	Abstract class; extends `Object`. Implements `Serializable` and `Comparable`.	Not related to enums but used with collections.

Enum with `switch` Statement

- Prior to Java 1.5:
  - Allowed types for `switch`: `byte`, `short`, `char`, `int`.
- From Java 1.5:
  - Added support for wrapper classes and enums.

Key Points:

- Case labels must match valid enum constants.
- Enum can be declared:
  - **Outside a class**:

        - Allowed modifiers: `public`, `default`, `strictfp`.
    - **Inside a class**:
        - Allowed modifiers: `public`, `private`, `default`, `protected`, `strictfp`, `static`.
    - **Not allowed inside methods**.

Enum and Inheritance

- In Java, every enum:
    - Directly extends `java.lang.Enum`.
    - Is implicitly `final` (cannot have child enums).
- **Conclusion**:
    - Explicit inheritance is not applicable for enums (no `extends` keyword).
    - However, enums can implement multiple interfaces.

Methods in `java.lang.Enum`

1. **`values()` Method**:

    - Lists all constants of the enum.
    - **Example**:

    ```
    Month[] months = Month.values();
    ```

2. **`ordinal()` Method**:

    - Retrieves the ordinal value (index) of an enum constant.
    - **Example**:

    ```java
    public final int ordinal();
    ```

Special Features of Java Enums

- Unlike older languages, Java enums support:
    - Variables, constructors, methods, etc.
- Enums can:
    - Declare a `main` method.
    - Be invoked directly from the command prompt.

Enum and Constructors

- Enums can have constructors.
- Characteristics:
    - Every enum constant represents an object of the enum class.
    - Enum constants are created at class loading time.
    - Constructors are executed during enum class loading.

Example:

```java
enum Color {
    RED, BLUE, GREEN;

    Color() {
        System.out.println("Constructor called for: " + this);
    }
}
```

This structured summary provides a comprehensive overview of the key concepts related to enums in Java. Let me know if you'd like additional formatting or examples.