**enums**

– enums define implicit methods such as

name()

ordinal()

– java.lang.Enum define a sole static method public static <T extends Enum<T>> T valueOf(Class<T> enumType, String name)

– enums define a constant T values[]

– enums in Java are type-safe (each enum has its own namespace)

– enum constants are implicitly static and final and cannot be changed once created

– enum can contain constructors, methods, variables, static and instance initers, and constant class bodies

– enum's body can be empty but if not, it must start with the declaration of constant(s)

– enum constructors can have parameters and support overloading.

– enum constructors can never be invoked directly in code; they are always called automatically whenever an enum is initialized.

– all enum constructors (including the default one) are implicitly private (even the ones that have no access modifier; as such, public and protected modifiers are not allowed) 🡪 it is impossible to create enum instances by using the new keyword 🡪 enum constants can only be created inside enums themselves.

– an enum instance is created when any enum constant is first called

– java.lang.Enum's sole constructor is

protected Enum(String name, int ordinal)

– enums can be compared using:

– switch

– ==

– equals()

– enum specifies a list of constant values assigned to a type.

– enums can be declared outside or inside a type (class, interface, or enum) but **NOT** in a method (even a static one), ctor, instance or static initers (comperr: 'enum types must not be local').

– enums can't be final or abstract regardless of location (outside or inside class)

– a top-level enum accepts only public or default 🡪 must NOT be marked static, final, abstract, protected, or private

– trailing comma after the list of constants is allowed: enum MyEnum { ONE, TWO, THREE, }

– empty stats (namely, semicolons) are allowed: enum Omg { ;;;;; }

– the list of constants must end with a semicolon when enum declares something else in addition to the constants

– enum declarations are allowed **only in static non-local context** 🡪 can't be declared as local vars inside methods + can't be declared inside ctors, instance or static initers or non-static inner classes (static classes are OK, though)

– switch case labels accept **simple** ('unqualified', as the compiler puts it) enum constant names only (i.e., case Seasons.WINTER: break; won't compile)

– UNLIKE regular classes, enum's static fields can't appear in enum's ctors or instance initers