Kotlin – Enum Classes

Enums seguem mesma estrutura conhecida do mundo Java. Cada constante de uma enum é um objeto. O Enum pode ser inicializado no construtor como nesse exemplo.

```
enum class Cor(val rgb: Int) {
VERMELHO(0xFF0000),

VERDE(0x00FF00),

AZUL(0x0000FF)

}
```

Um recurso interessante do Kotlin é que uma Enum pode implementar uma interface (porém não pode herdar de uma classe). Para isso, é necessário ela fornecer uma implementação do método da interface para cada entrada da enum.

```
enum class IntArithmetics : BinaryOperator<Int>, IntBinaryOperator {
PLUS {

override fun apply(t: Int, u: Int): Int = t + u

},

TIMES {

override fun apply(t: Int, u: Int): Int = t * u

};

     override fun applyAsInt(t: Int, u: Int) = apply(t, u)

}
```

Por default, enum em Kotlin oferece possibilidade de obter pelo nome e a lista de enum através de métodos da própria API nativa da linguagem. Caso uma busca por nome, seja de alguma constante não defiinda na Enum, um erro *IllegalArgumentException* é levantado.

```
EnumClass.valueOf(value: String): EnumClass
EnumClass.values(): Array<EnumClass>
```

As enums em Kotlin oferece acesso a duas propriedades, o nome e a posição dela dentro da declaração da Enum.

```
val name: String
val ordinal: Int
```

Enums, implementam Comparable e a ordem natural padrão é a ordem na qual ela foi declarada na classe que criou a enum.