2. Когда и зачем использовать перечисления

Основное правило — использовать enum, когда:

вы заранее знаете весь список значений;

эти значения — единственно допустимые для выбора.

Это могут быть дни недели, месяцы или любой другой **ограниченный** набор значений.

Перечисления не являются незаменимыми — иногда вместо них используют базы данных или обычные классы. Чтобы решить, нужен ли вам enum в каком-то конкретном случае, надо понимать разницу между перечислениями и классами.

Сходства и различия перечислений и классов

`Enum` имеет как сходства, так и различия с обычными классами.

Так же как и в обычные классы, в перечисления можно добавить методы, переменные и конструкторы. Кроме того, перечисления могут имплементировать интерфейсы.

```
// При создании перечисления указываем интерфейс,
// который необходимо имплементировать
public enum Currency implements Runnable {

   PENNY(1), NICKLE(5), DIME(10), QUARTER(25);
   private int value;

   Currency(int value) {
      this.value = value;
   }

   // Реализуем в enum'е абстрактный метод, объявленный в интерфейсе
   @Override
   public void run() {
      System.out.println("Перечисления могут реализовывать интерфейсы");
   }
}
```

Различия между перечислениями и классами

То, что мы используем именно **перечисления** в определенных ситуациях, продиктовано их **отличиями** от обычных классов:

> Перечисления не могут быть частью иерархии наследования и потому обладают особенным набором только им присущих методов.
>
> Их нельзя инстанцировать, и потому конструктор у перечисления может быть только приватный.

Далее мы поговорим подробно об этих особенностях перечислений: об уникальных методах и о том, как работать с ними, о нюансах работы с конструктором внутри перечислений, а также о том, как перечисления объявлять.