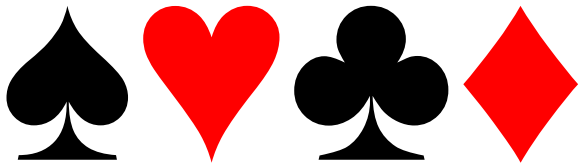
Object Oriented Programming with Java

PART 1: JAVA PROGRAMMING BASICS

Enums, By Aphrodice Rwagaju

Fixed values

- Sometimes a variable needs to represent a value which can be anything, e.g.
  - Height of a person
  - Weight of a person
- Other times a variable can only have one of a fixed set of values, e.g.
  - Gender ("male" or "female")
  - Card suits ("spades", "hearts", "diamonds", "clubs")



Fixed values

- We could represent these fixed values as booleans or integers

```
boolean gender; // True for MALE
                // False for FEMALE

int cardSuit;   // 1= spades
                // 2 = hearts, etc
```

- But this on its own is quite complicated and relies on the programmer remembering what each value means

Named values

- To improve things we could use named constants, e.g.

```
public static final int SUIT_SPADES = 1;
public static final int SUIT_HEARTS = 2;
public static final int SUIT_CLUBS = 3;
public static final int SUIT_DIAMONDS = 4;

int suit = SUIT_HEARTS;
```

- But there's nothing to stop `suit = 5` or `suit = -1` which are meaningless values

Enums

- Enum type is a special data type that enables for a variable to be a set of predefined constants
- Enums were added to Java in version 1.5
- They allow us to create variables which can only have a fixed set of values, e.g.

```
enum Suit { SPADES, HEARTS, CLUBS, DIAMONDS }

Suit suit1 = Suit.HEARTS;
Suit suit2 = Suit.CLUBS;
```

```
Suit suit3 = 1;
```

Not allowed!

Enums

- Enums are classes but they can't be instantiated...

```
Suit suit = new Suit();
```

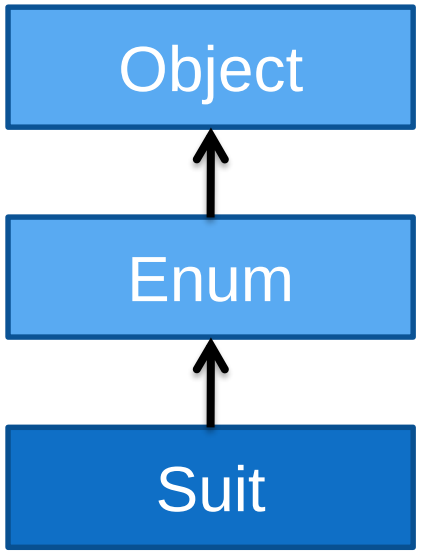
Not allowed!

- We can only reference the instances defined inside the enum...

```
Suit suit1 = Suit.HEARTS;
Suit suit2 = Suit.CLUBS;
```

Enums as classes

- All enums implicitly inherit from Enum and so have some methods built in...




```
Suit suit1 = Suit.HEARTS;

String n = suit1.name();
int o = suit1.ordinal();
```

Returns "HEARTS"

Returns 1, which is the index of the HEARTS value

Enum methods

- Being classes they can have methods
- They can override methods from `Object`, e.g.

```
public enum Suit {
   SPADES, HEARTS, CLUBS, DIAMONDS;

   public String toString() {
     switch (this) {
     case SPADES:
       return "Spades";
     case HEARTS:
       return "Hearts";
     case CLUBS:
       return "Clubs";
     default:
       return "Diamonds";
     }
   }
}
```

Enum methods

```
public enum UserRole {
	ADMINISTRATOR("Administrator"), EMPLOYEE("Employee"),
	PROCUREMENT("Procurement officer"),LOGISTICS("Logistics
	officer"), DAF("Director of Administration and Finance"),
	NONE("");

	private String roleDescription;

	UserRole(String roleDescription) {
		this.roleDescription = roleDescription;
	}
	public String getRoleDescription() {
		return roleDescription;
	}
}
```

Reference

https://docs.oracle.com/javase/tutorial/java/javaOO/enum.html

EoF