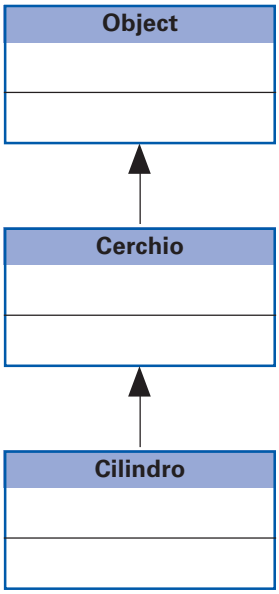
Casting tra le classi

Il **casting** è l'operazione che consente la conversione tra diversi tipi di dato predefiniti. È possibile applicare il *casting* anche alle classi: il **casting tra classi** è l'operazione con la quale un oggetto può cambiare classe se rispetta certe condizioni.

La creazione di un oggetto, usando l'operatore *new*, genera un'istanza di una classe che, in base al principio di ereditarietà, è anche un'istanza di tutte le sue sopraclassi. Esprimendo il concetto in modo più formale si ottiene la seguente **regola**:

> Avendo un oggetto d e due classi B, C
> SE (*B* è sottoclasse di *C*) e (*d* è un'istanza di *B*)
> ALLORA (*d* è anche un'istanza di *C*)

Per esempio, se si considera la gerarchia delle classi *Cilindro*, *Cerchio* e *Object*, si può dire che un'istanza della classe *Cilindro* è anche istanza della classe *Cerchio* e della classe *Object*, perché la classe *Cilindro* eredita tutti gli attributi delle sue sopraclassi.
Al contrario, un'istanza della classe *Cerchio* non può essere considerata anche istanza della classe *Cilindro*, perché gli oggetti della classe *Cilindro* hanno attributi e metodi aggiuntivi rispetto alla classe *Cerchio*.



Il **casting da una classe a una sopraclasse** è un'operazione consentita, per esempio è possibile trasformare un cilindro in un cerchio. L'operazione di *casting* viene eseguita inserendo tra parentesi tonde il nome della classe nella quale si vuole convertire un oggetto.

```
Cilindro cil = new Cilindro(4.0, 10.0);
Cerchio  cer;
Object   obj;

cer = (Cerchio) cil;
obj = (Object)  cil;
```

Le ultime due istruzioni effettuano il *casting* dell'oggetto *cil*. Con la prima si trasforma in un oggetto *cer* di classe *Cerchio*, con l'altra in un oggetto *obj* di classe *Object*.
Tenuto conto della strutturazione della gerarchia di classi di Java, a qualunque classe si può applicare il casting verso la classe *Object*.

Il **casting da una classe a una sottoclasse** è possibile solo se l'oggetto è veramente un esemplare della sottoclasse, altrimenti non è permesso. Il controllo della correttezza di questo casting può essere fatto solo durante l'esecuzione (*run-time*). Se il casting non è permesso, viene segnalata l'eccezione **ClassCastException**.

Il seguente esempio mostra un *casting* errato:

```
Cilindro cil;
Cerchio  cer = new Cerchio(4.0);

cil = (Cilindro) cer;
```

L'ultimo assegnamento provoca un errore, perché si tenta di fare il *casting* di un oggetto verso una sottoclasse.

Il seguente *casting* è invece corretto:

```
Cilindro cil1 = new Cilindro(4.0, 10.0);
Cilindro cil2;
Object   obj;

obj  = (Object) cil1;
cil2 = (Cilindro) obj;
```

La prima istruzione crea un oggetto *Cilindro* che viene riferito tramite *cil1*.

L'istruzione che esegue il casting da *cil1* a *obj* fa cambiare il modo con cui l'oggetto viene osservato.

L'oggetto resta sempre un esemplare della classe *Cilindro*, ma viene riferito tramite una variabile dichiarata di classe *Object*. Questo è lecito perché il *casting* verso le sopraclassi è permesso.

L'ultima istruzione esegue il *casting* inverso, dalla classe *Object* verso la sottoclasse *Cilindro*. Questo è possibile perché *obj* fa riferimento ad un oggetto che è un esemplare della classe *Cilindro*. Se così non fosse, cioè se *obj* si riferisse a un esemplare della classe *Object*, il casting non si potrebbe fare.