Ereditarietà

Ereditarietà

- Ereditarietà come meccanismo di estensione del comportamento
- Esempi:
  - Aggiungere funzionalità di colori ad una classe Finestra
  - Aggiungere capacità di ordinamento ad una classe Agenda
  - Aggiungere un *middle name* alla classe Name

- Modifichiamo la classe esistente ?
- **Potrebbe non essere desiderabile**
  - La classe è già rigorosamente verificata e robusta
  - Allargare la classe comporta una complessità aggiuntiva
  - Il codice sorgente potrebbe non essere disponibile
  - Le modifiche possono non essere consigliabili (ad esempio per le classi Java predefinite)

Ereditarietà

- E' un meccanismo per estendere classi esistenti, aggiungendo altri metodi e campi.

```
public class SavingsAccount extends BankAccount
   {
     nuovi metodi
     nuove variabili d'istanza
   }
```

- Tutti i metodi e le variabili d'istanza della classe **BankAccount** sono ereditati automaticamente
- Consente il riutilizzo del codice

Ereditarietà

- La classe preesistente (più generica) è detta SUPERCLASSE e la nuova classe (più specifica) è detta SOTTOCLASSE
  - **`BankAccount`**: superclasse
  - **`SavingsAccount`**: sottoclasse

Base comune a tutte le classi

- La classe `Object` è la superclasse di tutte le classi.
  - Ogni classe è una sottoclasse di `Object`

- Ha un piccolo numero di metodi, tra cui
  - `String toString()`
  - `boolean equals(Object otherObject)`
  - `Object clone()`

Diagramma di ereditarietà