Java, les objets : tout de suite !
Héritage

Cnam Paris
jean-michel Douin, douin@cnam.fr
Version du xx Mars 2003

Notes de cours associées au chapitre 8
tutorial BlueJ
http://www.bluej.org/doc/documentation.html

Ce support accompagne, référence le livre de David J. Barnes & Michael Kölling
Objects First with Java A Practical Introduction using BlueJ
Pearson Education, 2003 ISBN 0-13-044929-6.

Sommaire

- **Héritage et sous-classes**
- **Affectation polymorphe**
- **Liaison dynamique**

Héritage et Classification

- **Définir une nouvelle classe en ajoutant de nouvelles fonctionnalités à une classe existante**
  - **ajout de nouvelles fonctions**
  - **ajout de nouvelles données**
  - **redéfinition de certaines propriétés héritées (masquage)**
- **Une approche de la classification en langage naturel**
- Le verbre « être » : Les carrés **sont** des polygones réguliers ...

Un exemple : les étudiants de la Cité Descartes

- **Les étudiants de l 'ENPC**
- **Les étudiants de l 'ESIEE**
- **IUT,Université, ..**

- **Premier choix de conception:**

  une classe par étudiant de chaque école

  EtudiantENPC,

  EtudiantESIEE

  ....

La classe EtudiantENPC

```
public class EtudiantENPC{
  private String nom;
  private String prénom;
  private int    promotion;

  public EtudiantENPC(String nom, String prénom, int promotion){
    this.nom       = nom; this.prénom = prénom;
    this.promotion = promotion;
  }
  public String loginENPC(){
    // algorithme propre à l 'école
  }

  public String getNom(){       return nom;  }
  public String getPrénom(){    return prénom;  }
  public String toString(){
    return getNom() + "_" + getPrénom() + ":" + promotion;
  }
}
```

La classe EtudiantESIEE

```
public class EtudiantESIEE{
  private String nom;
  private String prénom;
  private int    année;

  public EtudiantESIEE (String nom, String prénom, int année){
    this.nom       = nom; this.prénom = prénom;
    this.année = année;
  }
  public String loginESIEE (){
    // algorithme propre à l 'école
  }

  public String getNom(){       return nom;  }
  public String getPrénom(){    return prénom;  }
  public String toString(){
    return loginESIEE();
  }
}
```

Les étudiants de la cité Descartes

```
public class CitéDescartes{
  private ArrayList listeDesEtudiants;
  ….
  public CitéDescartes(){
    listeDesEtudiants = new ArrayList();
  }


  public void ajouterUnEtudiantESIEE(EtudiantESIEE e){
    listeDesEtudiants.add(e);
  }

  public void ajouterUnEtudiantENPC(EtudiantENPC e){
    listeDesEtudiants.add(e);
  }

...
```

Premier Bilan

- **Similitudes nombreuses entre les classes Etudiant**

- **Une nouvelle Ecole engendre :**
  - **Une nouvelle classe Etudiant**
  - **Nouvelle implémentation de la classe CitéDescartes**

```
public void ajouterUnEtudiantIUT(EtudiantIUT e){
   listeDesEtudiants.add(e);
}
```

EtudiantENPC, EtudiantESIEE

- **similitudes :**
- **champs d 'instance communs**

```
private String nom;
private String prénom;
```

- **méthodes communes**

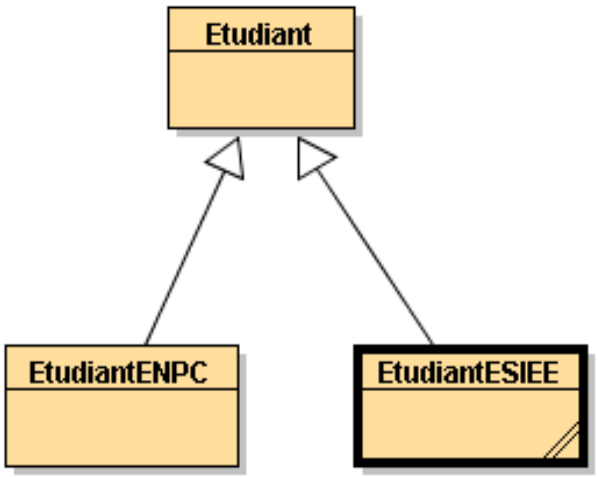
```
public String getNom(){        return nom;  }
public String getPrénom(){     return prénom;  }
```

or...

Les étudiant de l 'ESIEE sont des **étudiants**

Les étudiants de l 'ENPC sont des **étudiants**

Etudiant




```
public class EtudiantENPC extends Etudiant{
```

La classe Etudiant

```
public class Etudiant{
  private String nom;
  private String prénom;

  public Etudiant (String nom, String prénom){
    this.nom       = nom;
    this.prénom = prénom;
  }

  public String getNom(){       return nom;  }
  public String getPrénom(){    return prénom;  }
}
```

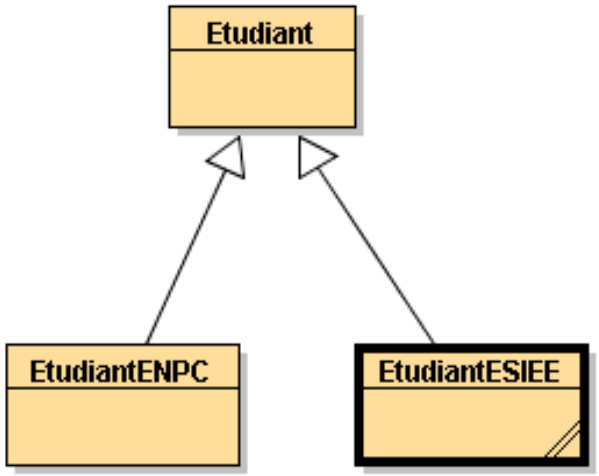
La classe EtudiantENPC

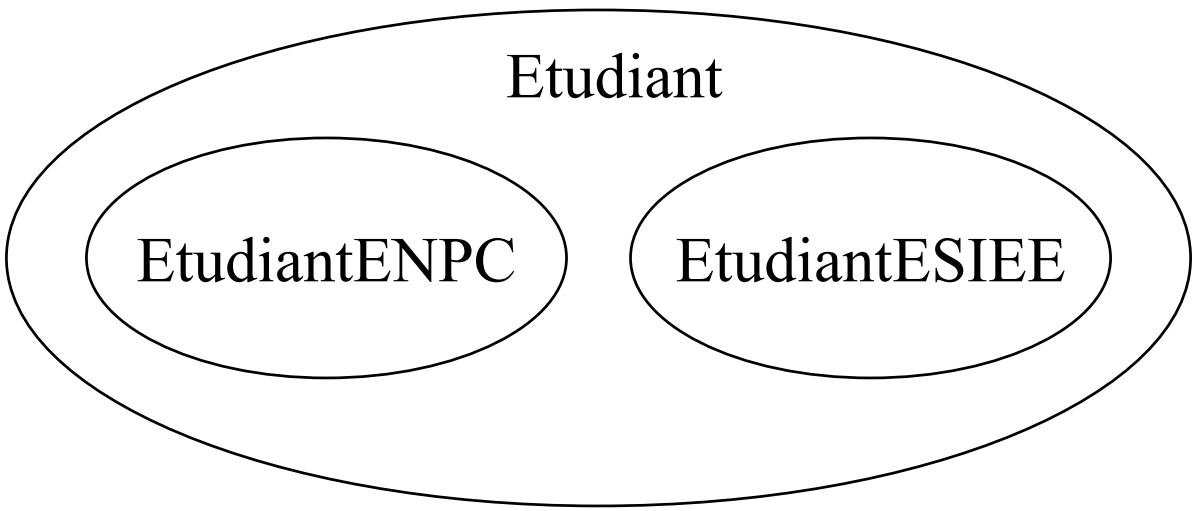
```
public class EtudiantENPC extends Etudiant{
private int    promotion;

  public EtudiantENPC(String nom, String prénom; int promotion){
    super(nom,prénom);
    this.promotion = promotion;
  }
  public String loginENPC(){
    // algorithme propre à l 'école
  }

 public String toString(){
    return getNom() + "_" + getPrénom() + ":" + promotion;
  }
}
```

Graphe d ’héritage





Instances et Affectation

- **Etudiant e;**
- **e = new EtudiantENPC("alfred", "paul", 05);**
- **e = new EtudiantESIEE("paul", "alfred", 2001);**
- **e = new Etudiant("alfred", "paul");**

- **???**
- **EtudiantESIEE ee = e;**

Avantages escomptés

- **Évite la duplication de code**
- **Réutilisation accrue**
- **Maintenance facilitée**

Les étudiants de la cité Descartes

```
public class CitéDescartes{
  private ArrayList listeDesEtudiants;
  ….
  public CitéDescartes(){
    listeDesEtudiants = new ArrayList();


  //public void ajouterUnEtudiantESIEE(EtudiantESIEE e){
  //public void ajouterUnEtudiantENPC(EtudiantENPC e){
  // sont remplacés par


  public void ajouterUnEtudiant(Etudiant e){
    listeDesEtudiants.add(e);
  }
```

Instances et Affectation (bis)

- **Etudiant e = new EtudiantESIEE(…);**
- **EtudiantESIEE ee = e; // erreur de compilation**
- **EtudiantESIEE ee = (EtudiantESIEE)e; // ok**
- **...**

Héritage en Java

- **les champs d ’instances sont cumulés**
- **Les méthodes de la super classe peuvent être re-définies**
  - **Même nom, même signature**

super(), super.

- **super()**

  appel du constructeur de la super classe
  en première ligne de code

```
public class EtudiantENPC extends Etudiant{
  private int    promotion;
  public EtudiantENPC(String nom, String prénom; int promotion){
    super(nom,prénom);
    this.promotion = promotion;
}
```

- **super.**

  Sélection d ’un champ d ’instance (si accessible)
  Sélection d ’une méthode de la super classe

super.

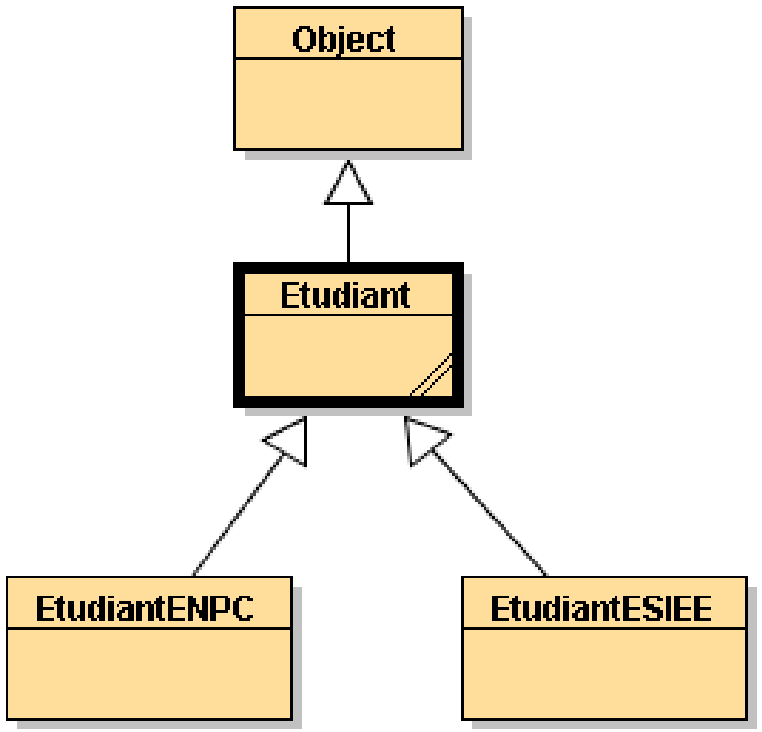
```
public class EtudiantENPC extends Etudiant{

   private int    promotion;

   public EtudiantENPC(String nom, String prénom; int promotion){
     super(nom,prénom);
     this.promotion = promotion;
   }

   // redéfinition de getNom()
   public String getNom(){
     return super.getNom().toUpperCase();
   }
```

La classe Object : racine de toute classe en Java



Instances et Affectation (ter)

- **Object o;**
- **o = new EtudiantESIEE(…);**
- **System.out.println(o.toString());**

- **type déclaré**
- **type constaté**

o est de type déclaré Object
o est de type constaté EtudiantESIEE

Résumé, synthèse