NJUSWI
2002-2012
Anniversary
南京大學 软件学院

封装

刘 钦

南京大学软件学院

需求的变更

- Unit
  - Kilometers
  - NauticalMiles
  - StatueMiles
  - Radians
- Three implementations for geometry
  - PlaneGeometry
  - SphericalGeometry
  - EllipticalGeometry

Isolating potential change

将需求的变更封装在类里面！

```
public class Position
{
  public Position( double latitude, double longitude )
  {
    setLatitude( latitude );
    setLongitude( longitude );
    // Default to plane geometry and kilometers
    geometry = new PlaneGeometry();
    units = new Kilometers();
  }
  public void setLatitude( double latitude )
  {
    setPhi( Math.toRadians( latitude ) );
  }
  public void setLongitude( double longitude )
  {
    setTheta( Math.toRadians( longitude ) );
  }
  public void setPhi( double phi )
  {
    // Ensure -pi/2 <= phi <= pi/2 using modulo arithmetic.
    // Code not shown.
    this.phi = phi;
  }
  public void setTheta( double theta )
  {
    // Ensure -pi < theta <= pi using modulo arithmetic.
    // Code not shown.
    this.theta = theta;
  }
  // Setters for geometry and units not shown
```

Good Design

```
  public double getLatitude()
  {
    return( Math.toDegrees( phi ) );
  }
  public double getLongitude()
  {
    return( Math.toDegrees( theta ) );
  }
  // Getters for geometry and units not shown
  public double distance( Position position )
  {
    // Calculate and return the distance from this object to
    // position using the current geometry and units.
  }
  public double heading( Position position )
  {
    // Calculate and return the heading from this object to
    // position using the current geometry and units.
  }
  private double phi;
  private double theta;
  private Geometry geometry;
  private Units units;
}
```

```
Position myHouse = new Position( 36.538611, -121.797500 );
Position coffeeShop = new Position( 36.539722, -121.907222 )
double distance = myHouse.distance( coffeeShop );
double heading = myHouse.heading( coffeeShop );
System.out.println
  ( "Using " + myHouse.getGeometry() + " geometry, " +
    "from my house at (" +
    myHouse.getLatitude() + ", " + myHouse.getLongitude() +
    ") to the coffee shop at (" +
    coffeeShop.getLatitude() + ", " + coffeeShop.getLongitud
    ") is a distance of " + distance + " " + myHouse.getUnit
    " at a heading of " + heading + " degrees."
  );
myHouse.setGeometry( Geometry.SPHERICAL );
myHouse.setUnits( Units.STATUTE_MILES );
distance = myHouse.distance( coffeeShop );
heading  = myHouse.heading(  coffeeShop );
System.out.println
  ( "Using " + myHouse.getGeometry() + " geometry, " +
    "from my house at (" +
    myHouse.getLatitude() + ", " + myHouse.getLongitude() +
    ") to the coffee shop at (" +
    coffeeShop.getLatitude() + ", " + coffeeShop.getLongitud
    ") is a distance of " + distance + " " + myHouse.getUnit
    " at a heading of " + heading + " degrees."
  );
```

完备性

- 例子
  - 一个只能加水而不能倒水的杯子
  - 只能入库，不能出库的仓库

Encapsulation

- **Encapsulation rule 1:**
  - **Place data and the operations that perform on that data in the same class**
  - 将数据和操作数据的行为放在一起
- **Encapsulation rule 2:**
  - **Use responsibility-driven design to determine the grouping of data and operations into classes**
  - 用职责驱动的设计原则来决定数据和行为的在一起
- **Encapsulation rule 3:**
  - **The responsibility should be complete**
  - 职责要完备

类的职责与封装

- 数据职责
  - 表征对象的本质特征
  - 行为（计算）所需要的数据
    - 教务系统中学生对象：计算年龄
    - 税务系统中纳税人：计算所得税
- 行为职责
  - 表征对象的本质行为
  - 拥有数据所应该体现的行为
    - 出生年月
    - 个人收入

数据职责与行为职责“在一起”