Recommended Reading

- `http://www.gamespy.com/articles/devweek_b.shtm`, "A Critical Look at Programming Languages" by Tim Sweeney, principle architect of *Unreal*.

What is null?

```
/*
 * s is given the value null (similar to nil in Scheme).
 */
AShape s ;
```

- When a variable of some class is declared without any initial instantiation, it is *automatically* assigned the value null.

- null represents "non-existence". Assign null to a variable of some class only if you want to express "non-existence".

Instance Variables and Instance Methods

```
AShape s = new Rectangle(6, 7);

AShape t = new Rectangle(3, 4);

s.dArea();
```

- The code for dArea is executed and can only access the _dWidth and _dHeight of s. It does not know anything about the _dWidth and _dHeight of t.

- _dWidth and _dHeight are said to be *instance* fields (or variables) of Rectangle.

Instance Variables and Instance Methods (cont.)

- `double dArea()` is said to be an *instance* method of s of Rectangle.

  – It can only be called on an existing instance of a class.

Static Variables and Static Methods

- Suppose we want to keep track of how many Rectangles are being created during the course of our program. What do we need?

- Answer: a field that is unique and global to all instances of Rectangle and that can be accessed by all methods in Rectangle. In Java, we use the keyword static.

Static Variables and Static Methods (cont.)

```
class Rectangle extends AShape
{
    private static int _iCOUNT; // Initial value is 0.

    // ... other code ...

    public Rectangle(double width, double height)
    {
        // Each time this constructor is called,
        // _iCOUNT is incremented by 1.
        _iCOUNT++;
        // ... other code ...
    }
}
```
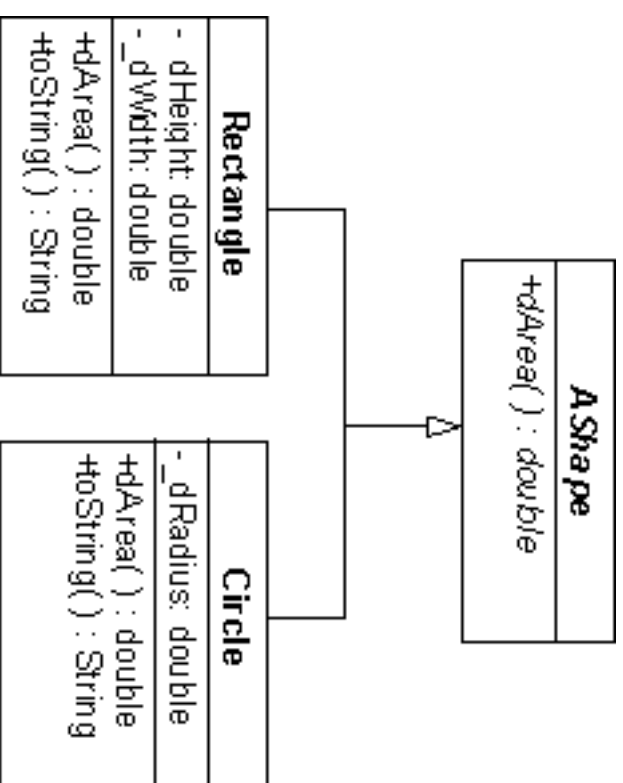
Static Variables and Static Methods (cont.)

- `static` methods cannot access instance fields. They can only access `static` fields.

- `static` methods can be called before any instantiation of the class. (Recall `main`.)

Example: `System` is a `class`. `out` is a `static` variable.

`System.out.println(cirPizza);`

Taxonomy Tree

```
            ┌──────────────────┐
            │     AShape       │
            ├──────────────────┤
            ├──────────────────┤
            │ +dArea() : double│
            └──────────────────┘
                    △
           ┌────────┴────────┐
┌──────────────────┐   ┌──────────────────┐
│    Rectangle     │   │      Circle       │
├──────────────────┤   ├──────────────────┤
│ - dHeight: double│   │ - dRadius: double │
│ - dWidth: double │   ├──────────────────┤
├──────────────────┤   │ +dArea() : double │
│ +dArea() : double│   │ +toString() : String│
│ +toString() : String│ └──────────────────┘
└──────────────────┘
```

- A class and all of its subclasses form a taxonomy tree. The above UML diagram shows the taxonomy tree of AShape and its variants (or subclasses), namely Rectangle and Circle.

Polymorphism

`AShape s = new Circle(2.7); // OK.`

`AShape t = new Rectangle(3, 4); // OK.`

`t = s; // OK, the old Rectangle is gone.`

`Circle u = new Rectangle(5, 6); // NO!`

- A variable of class AShape can be assigned any instance of subclasses of AShape at any time in a program. AShape is said to be *polymorphic*.

Polymorphism (cont.)

- In general, a variable of a superclass can be assigned an instance of any of its subclasses, but not the other way around. Polymorphism means a class can be represent any of its subclasses.

- *We can think of polymorphism as viewing the taxonomy tree from the top down.*

Inheritance

- The "is-a" relationship between two classes is called *inheritance*.

  - Class B *inherits* class A means that B can automatically access all the non-private fields of A and perform all the non-private methods of A.
    * In Java, B can override a non-private method of A, and provide new behavior by redefining it with new code.
    * B may also have additional fields and methods that A has absolutely no knowledge of. As such, B is said to be a specialization of A.

- *We can think of inheritance as viewing the taxonomy tree from the bottom up.*