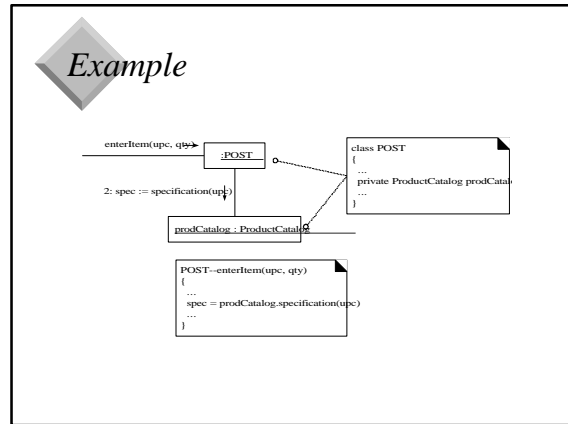


Visibility (Ch. 20)

- ❖ The ability of one object to “see” or have reference to another object
 - a scalar variable name is a type of reference
 - a reference can also be a pointer
- ❖ The UML has special notation for designating visibility
- ❖ Visibility is related to scope



Scope Example

```

int a = 1, b = 2;

void main()
{
    int b = -2,
    int c = 3;

    if ( . . . )
    {
        int c = -3;
        int d = 4;
        print(a,b,c,d);
    }

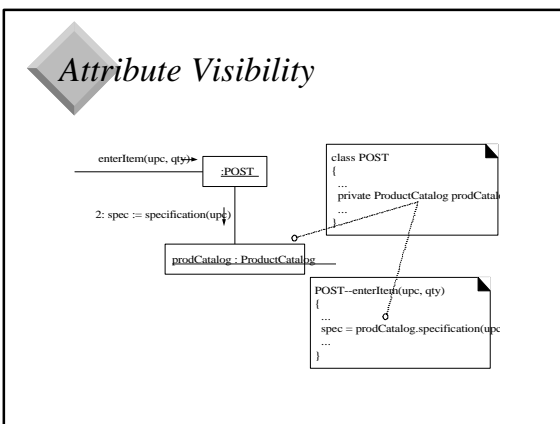
    print(a,b,c);
}
    
```

c = -3
d = 4
b = -2
c = 3
a = 1
b = 2

Run-time Stack

Common Visibility Categories

- ❖ Attribute Visibility
 - Objects can “see” their attributes
- ❖ Parameter Visibility
 - Parameters are pre-set local variables
- ❖ Locally Declared Visibility
 - Variable declared in the local scope
- ❖ Global Visibility
 - The outer-most scope, common to all modules

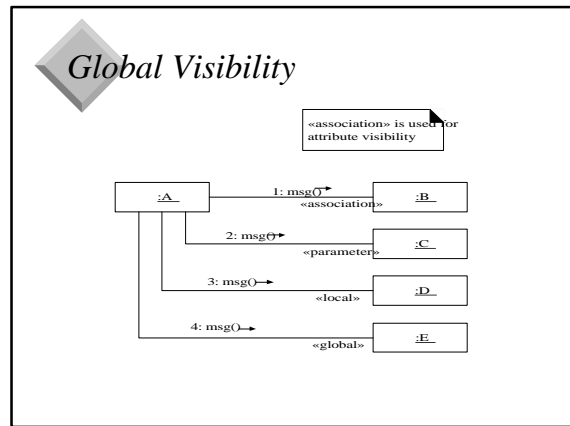
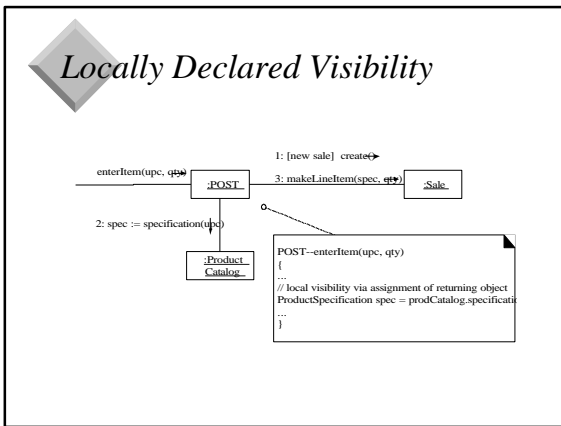
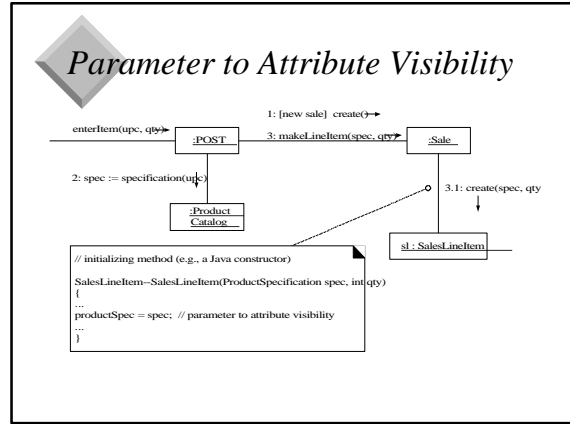
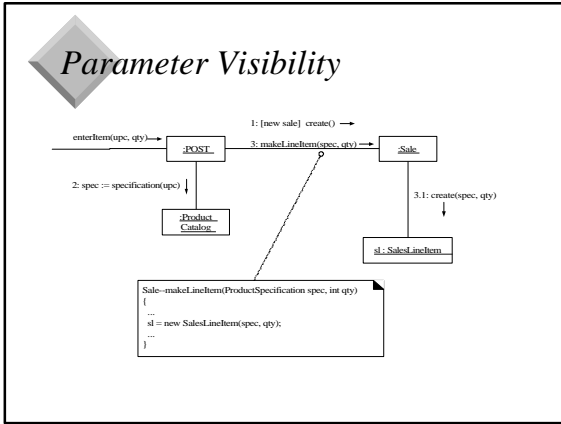


Parameter Visibility (C++)

```

void main()
{
    int a[10];
    ...
    swap(a[i], a[j]);
    ...
}

void swap(int& x, int& y)
{
    int t = x;
    x = y;
    y = t;
}
    
```



- ### Access Control
- ❖ Also known as *Permissions* or *Protection*
 - ❖ Public - visible to all outside the class
 - ❖ Private - visible only to members of the class
 - ❖ Protected - visible only to class and its descendants (requires inheritance)
 - ❖ Package (Implementation) - visible only to the class and other classes in its package

- ### Use of static and const
- ❖ Orthogonal concept to access control
 - ◆ parallel to, in conjunction with,
 - ❖ static - when used in a class declaration, means the member belongs to the class, not the instances
 - Hence no objects of the class are needed to invoke the member function
 - static methods can only access static data inside the class

Static Data Member (C++)

```
class Bunny
{
    static int how_many; ...
};
int Bunny::how_many = 0;
```

Maintain count in constructor & destructor

```
Bunny::Bunny()
{ how_many++; ... }

Bunny::~Bunny()
{ how_many--; ... }
```

Static Member Function (C++)

```
class Bunny
{ static int how_many; ...
public:
    static int HowMany()
    { return how_many; }
};
```

Invoke function with or without object

```
Bunny bunny1;
int n;
n = bunny1.HowMany();
n = Bunny::HowMany();
```

