THE UNIVERSITY OF MELBOURNE

SWEN30006
Software Modelling and Design

DESIGNING FOR VISIBILITY

Larman Chapter 19

A mathematician is a device for turning coffee into theorems.

—Paul Erdös

Objectives

On completion of this topic you should be able to:

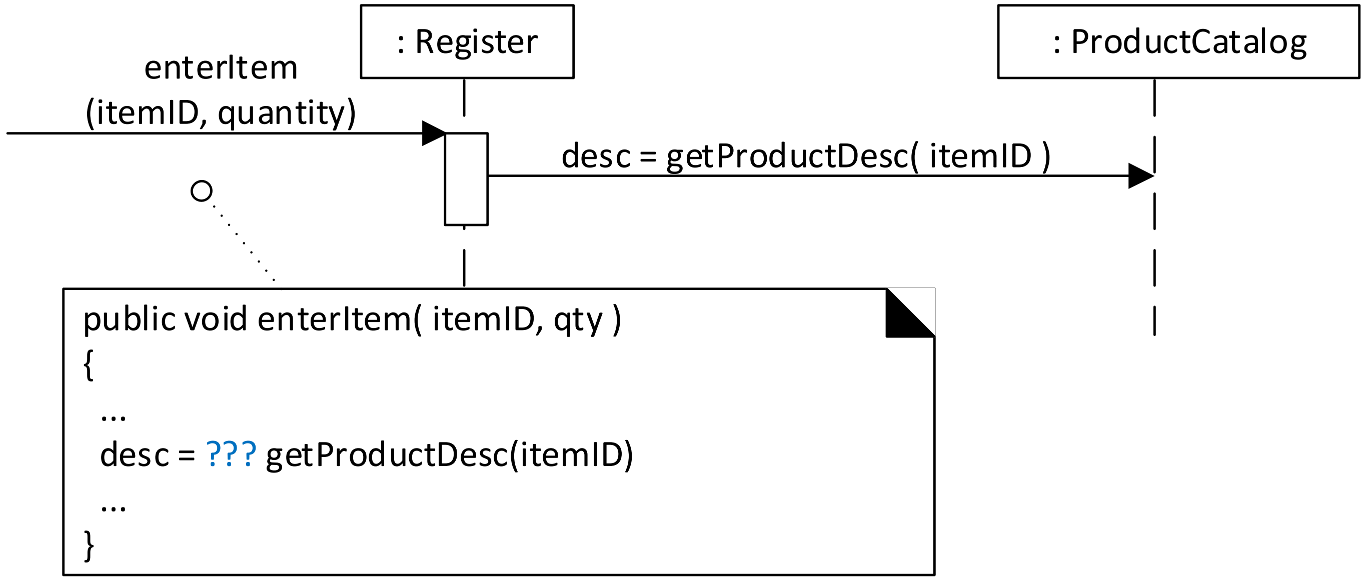
- Identify four types of visibility.
- Design to establish visibility where required.

Context

- OOAD applies encapsulation: objects and classes
- High cohesion encourages grouping highly related responsibilities
- Low coupling encourages minimising dependency on other elements
- Design requires assigning responsibilities and objects cooperating to achieve required outcomes
- Objects require visibility of each other to cooperate

Visibility from Register to ProductCatalog

How does Register gain visibility of Product catalog?



```
public void enterItem( itemID, qty )
{
  ...
  desc = ??? getProductDesc(itemID)
  ...
}
```

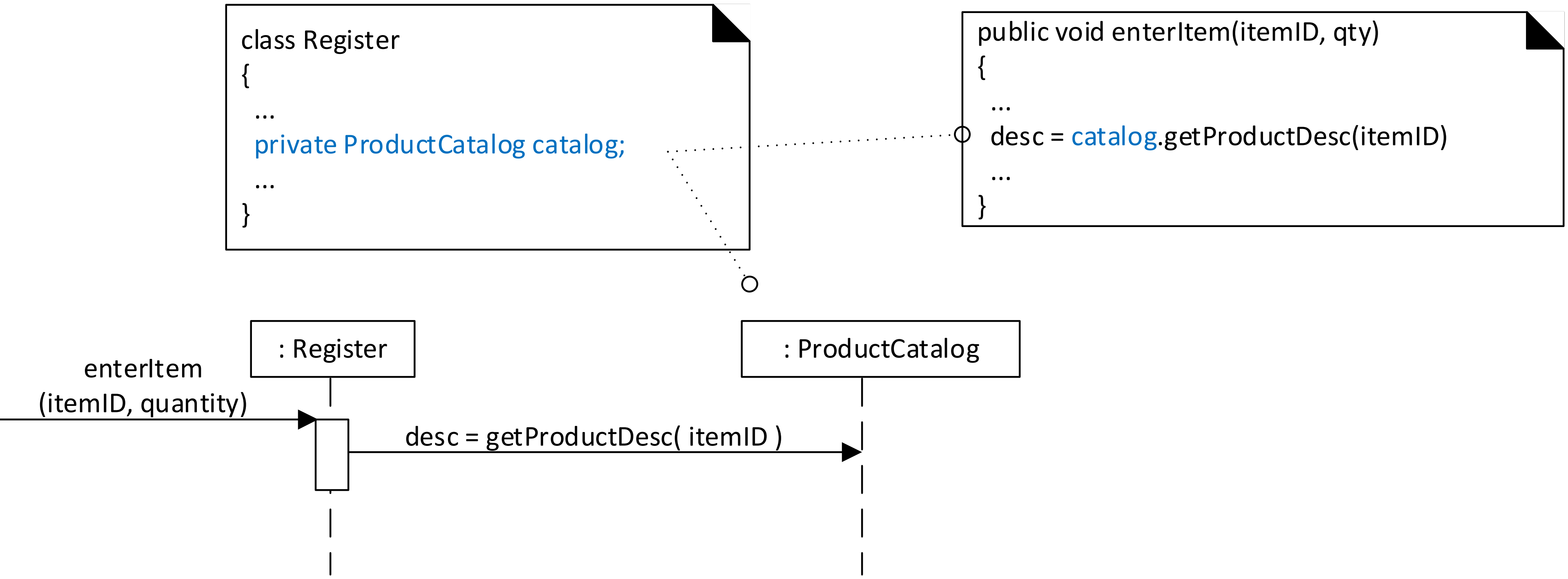
What is Visibility?

- Ability of an object to "see" or refer to another object
- For A to send a message to B, B must be visible to A

Is it in scope? Four common ways:

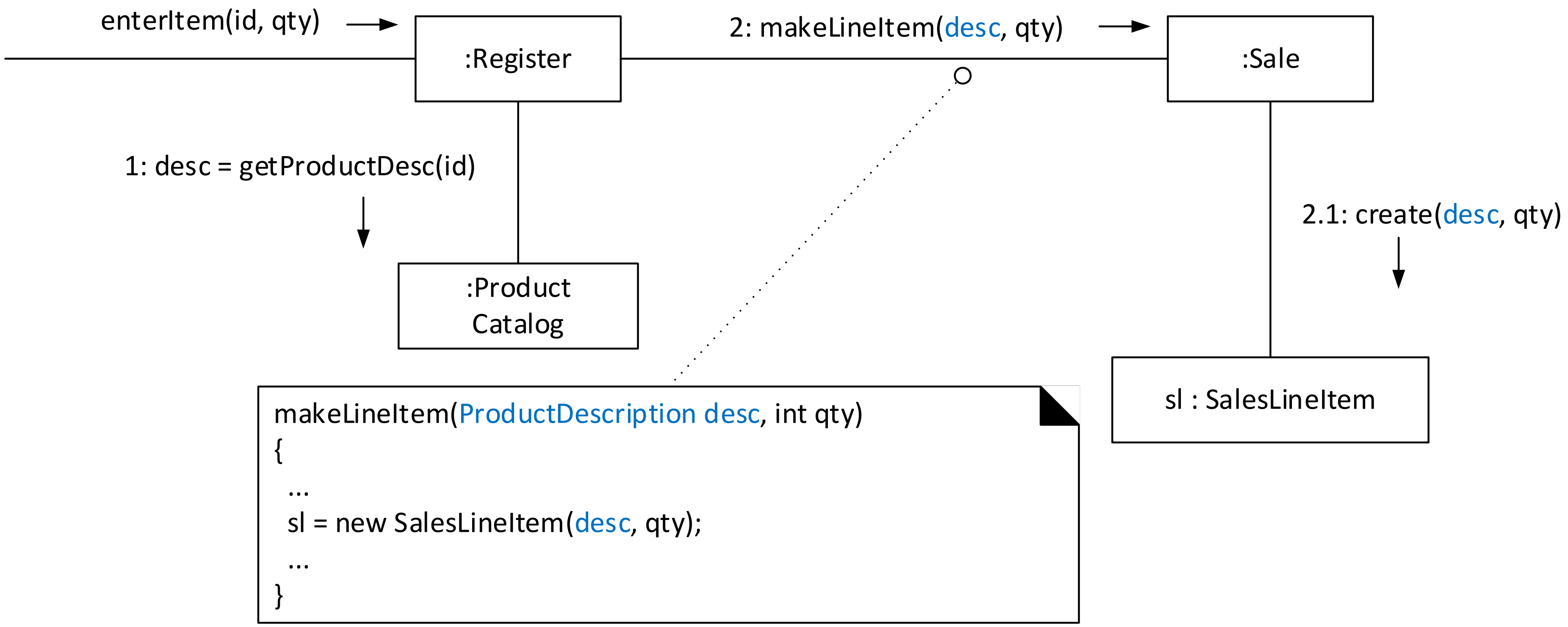
1. B is an *attribute* of A.
2. B is a *parameter* of a method of A.
3. B is a (non-parameter) *local* object in a method of A.
4. B has in some way *global* visibility.

1. Attribute Visibility


```
class Register
{
  ...
  private ProductCatalog catalog;
  ...
}
```

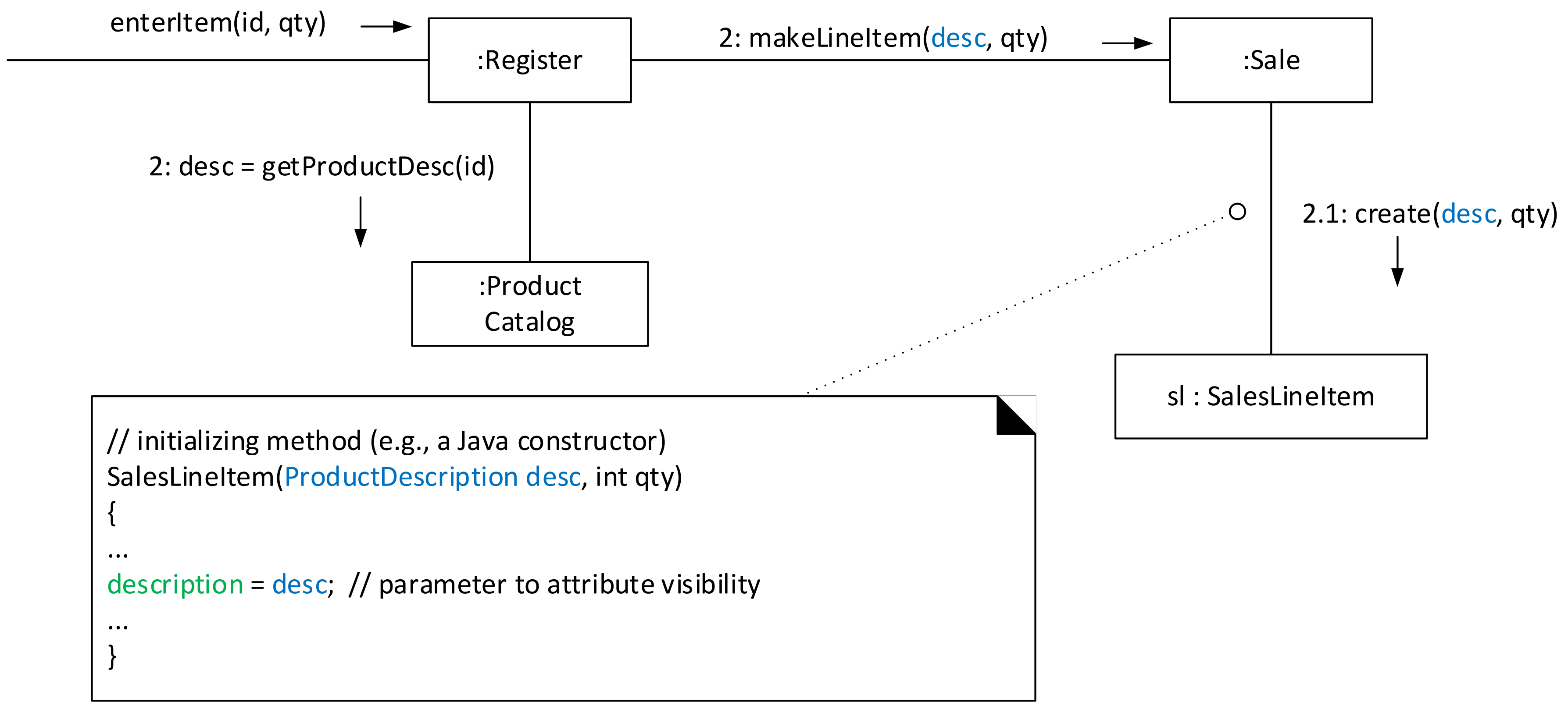
```
public void enterItem(itemID, qty)
{
  ...
  desc = catalog.getProductDesc(itemID)
  ...
}
```

2. Parameter Visibility



```
makeLineItem(ProductDescription desc, int qty)
{
  ...
  sl = new SalesLineItem(desc, qty);
  ...
}
```

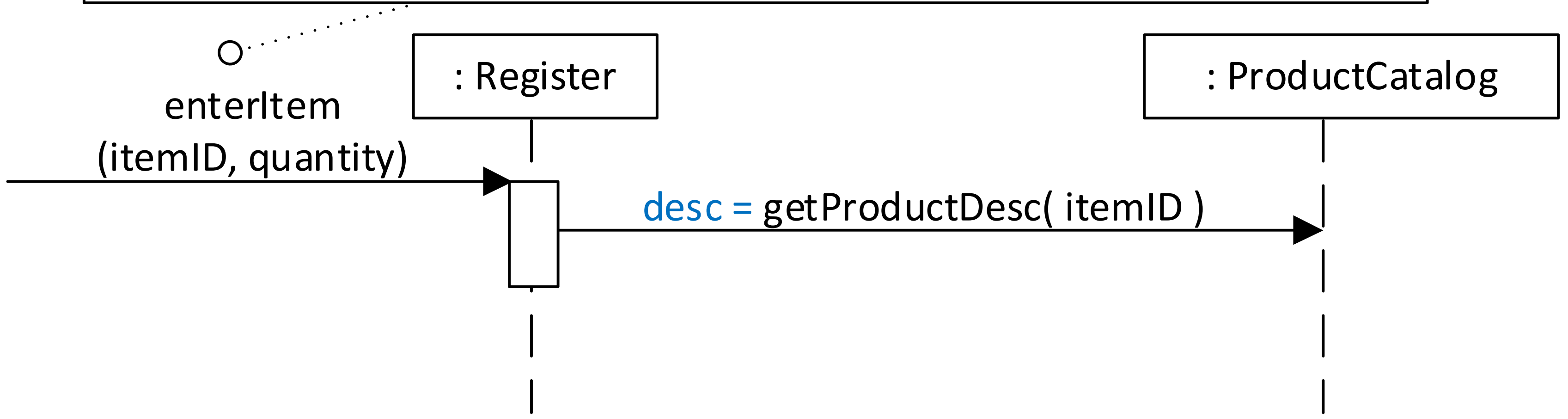
Parameter to Attribute Visibility



```
// initializing method (e.g., a Java constructor)
SalesLineItem(ProductDescription desc, int qty)
{
...
description = desc;  // parameter to attribute visibility
...
}
```

3. Local Visibility

```
enterItem(id, qty)
{
...
// local visibility of ProductDescription via assignment of returning object
ProductDescription desc = catalog.getProductDes(id);
...
}
```



```
// Cf. implicit local Catalog (not explicitly assigned)
ProductDescription desc = other.getCatalog().getProductDes(id);
```

4. Global Visibility

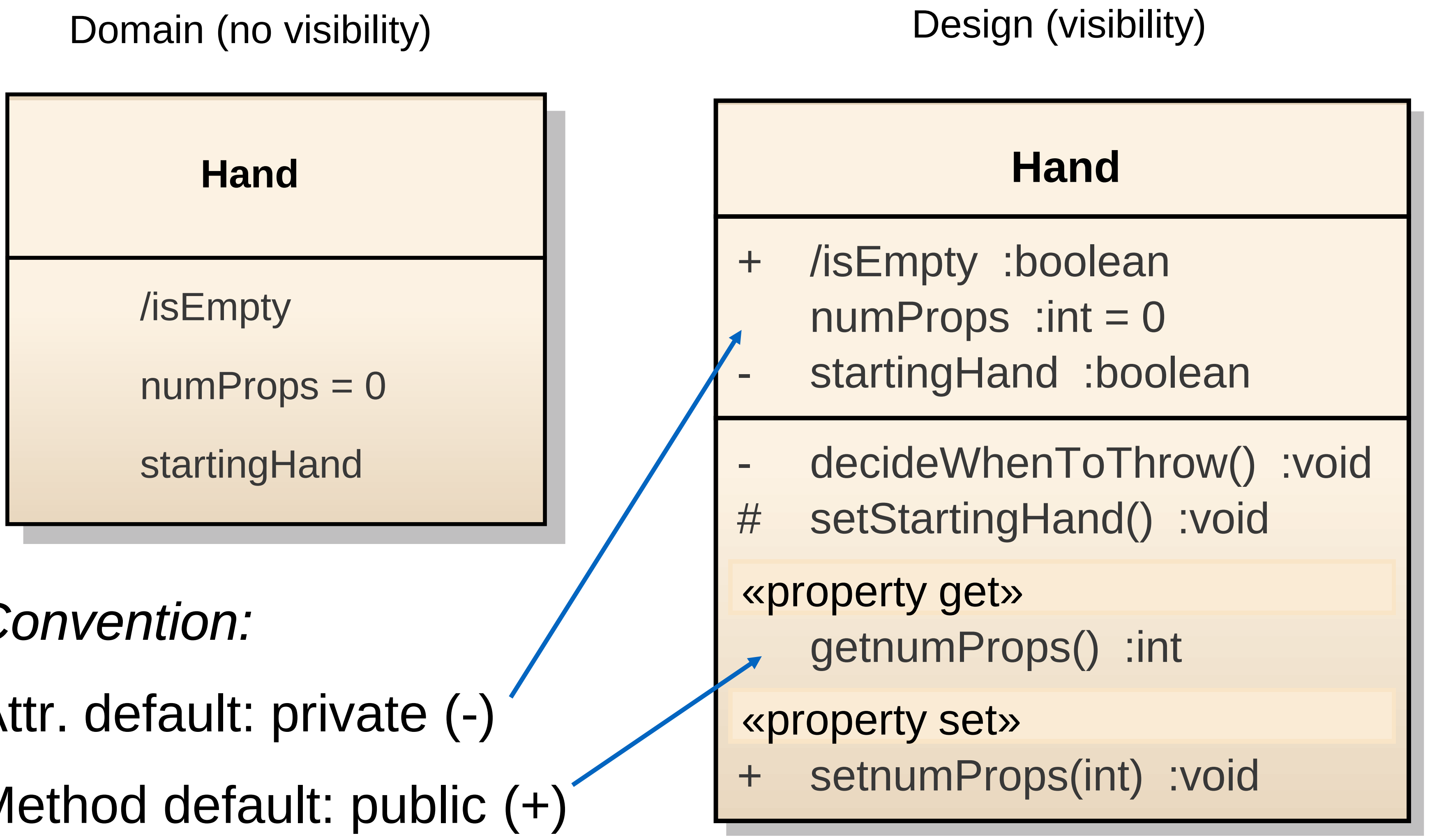
	C++	Java

Declaration	`int v = 1;`	`public class Global {`
		`    public static int v = 1;`
		`}`

Usage	`… main(…) {`
	
	`  int n1 = v;`
	`  int n2 = ::v;`
	`}`
		`… main(…) {`
		`  Global g = new Global();`
		`  int n1 = g.v;`
		`  int n2 = Global.v;`
		`}`

- Strictly, Java does not have global visibility
- Effect is achievable, but use Singleton (Ch. 26)

UML Visibility Marks



Convention:

Attr. default: private (-)

Method default: public (+)

Summary

- Objects need visibility of other objects to collaborate
- To be visible, an object must be in scope:
  - attribute; parameter; local; global
- Visibility marks (UML) or modifiers (Java) must allow access to required elements, and should restrict access where not required
- Choices about how to provide visibility impact on coupling