Object Relationships: 3 basic types:

Collaborate (uses a)

Composition (has a). *Aggregation* is a special type of composition where the component parts don’t exist unless as part of the composition.

Inheritance (is a)

Collaboration (uses a):

Exists when an object uses 1 or more instances of another class. Providing services to each other.

You can see this relationship when a class uses a seemingly unrelated class to perform an operation.

Composition (has a):

Usually when a type is a property.

List<> doesn’t have a good default value. Use *new* *List<Address>();*

To ctor chain – public Customer() : this(0)

*IEnumerable* is the recommended way to return a collection as the returnable value is more flexible for the caller. *IEnumerable<Address>*

You can use *ToList()* to turn an IEnumerable into a list.

CustomerRepository creates instances of Customer and AddressRepository in a collaborator relationship to return an instance of Customer.

Composition by ID is a relationship that doesn’t use object references but ID’s to reference an object. It can be more efficient. Loading all data isn’t required.

Make property ID’s for required data instead of directly referencing an object.

Composition by ID reduces coupling by saving the need to reference an object. The ID can be used to retrieve data.

Composition by ID means there might be the need for more classes to make the data usable.

Repository classes populate classes from a data store.

Order display class contains only the properties it needs to display the order.

Inheritance chaining – inheritance hierarchy through many classes. Similar to every class inheriting from Object. Lower objects in the chain are specialised.

Composition is achieved with a property or ID (references to the object).