Aggregation

In this lesson, we'll learn a new way of linking different classes.

WE'LL COVER THE FOLLOWING

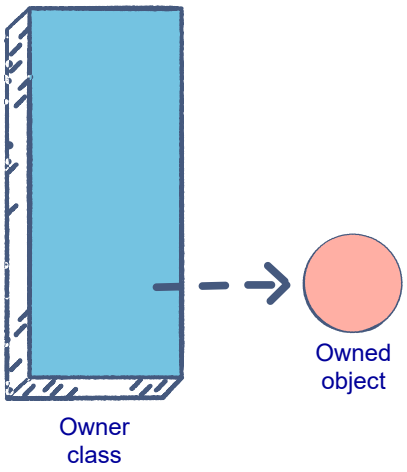
Aggregation is very similar to composition. It also follows the **Has-A** model. This creates a parent-child relationship between two classes, with one class owning the object of another.

So, what makes aggregation unique?

Independent Lifetimes

> In aggregation, the lifetime of the owned object does not depend on the lifetime of the owner.

The owner object could get deleted but the owned object can continue to exist in the program. In composition, the parent **contains** a child object. This bounds the child to its parent. In aggregation, the parent only contains a **reference** to the child, which removes the child's dependency.



The owner (parent) simply points to the owned object (child)

Aggregation

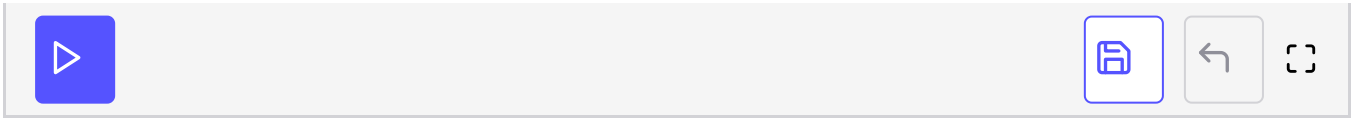
You can probably guess from the illustration above that we'll need pointers to implement aggregation.

Example

Let's take the example of people and their country of origin. Each person is associated with a country, but the country can exist without that person:

```
#include <iostream>
#include <string>
using namespace std;

class Country{
  string name;
  int population;

  public:
  Country(string n, int p){
    name = n;
    population = p;
  }
  string getName(){
    return name;
  }
};

class Person {
  string name;
  Country* country; // A pointer to a Country object

  public:
  Person(string n, Country* c){
    name = n;
    country = c;
```

```
  }

  string printDetails(){
    cout << "Name: " << name << endl;
    cout << "Country: " << country->getName() << endl;
  }
};

int main() {
  Country* country = new Country("Utopia", 1);
  {
    Person user("Darth Vader", country);
    user.printDetails();
  }
  // The user object's lifetime is over

  cout << country->getName() << endl; // The country object still exists!
}
```



As we can see, the `country` object lives on even after the `user` goes out of scope. This creates a looser relationship between the two in comparison to composition.

In the next lesson, we will explore the third type of linkage between classes; association.