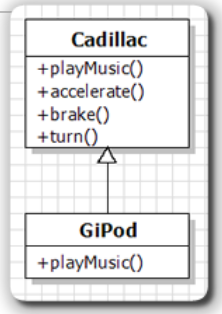
Private Inheritance



Private inheritance is one way to solve this problem. Private inheritance means you want to **inherit the implementation of a class, not the interface.** Use this when a class has some functionality that you want to exploit, but you don't want to use the interface of the base class.

To use private inheritance,replace the keyword **`public`** with **`private`**, or, simply omit it altogether. I'd recommend that you explictly specify **`private`**to prevent subsequent maintenance programmers from "fixing" your code inadvertently.

```
class GiPod : private Cadillac
{
    // ...
};
```

`GiPod` objects are not, in the **is-a** sense, **`Cadillac`**. Calling any "inherited" member functions will fail. To call any of the inherited member functions, you must add those member functions to the new interface, with a **`using`** declaration like this:

```
class GiPod : private Cadillac
{
public:
    using Cadillac::playMusic;
};
```

When you "import" a member function like this, you don't need to specify the arguments or supply an argument list. You do need the **`public:`** if you want the name moved into the **`public`** section.