either a property or a function), and if the member does not exist on the child object, it is
copied to the child object.

Using the createInheritance function is rather trivial: first create an instance of the child
object, and then use the createInheritance function, passing to it the child object and an
instance of the parent object, like so:

```javascript
var child = new Child();
createInheritance(new Parent(), child);
```

All the properties and methods on the parent object that don’t exist on the child object
will be copied to the child object.

**Putting It All Together**

You’ve now seen how private properties are possible in JavaScript and how JavaScript can sup-
port a more class-based approach to inheritance like C++ and Java. To demonstrate how it all
works, we’ll show how to convert the earlier example that used the Vehicle, SportsCar, and
CementTruck objects to use the new pattern of information hiding and inheritance. Listing 5-5
lists the new object definitions.

**Listing 5-5. classicalInheritance.js**

```javascript
function Vehicle() {
    var wheelCount = 4;
    var curbWeightInPounds = 4000;

    this.getWheelCount = function() {
        return wheelCount;
    }

    this.setWheelCount = function(count) {
        wheelCount = count;
    }

    this.getCurbWeightInPounds = function() {
        return curbWeightInPounds;
    }

    this.setCurbWeightInPounds = function(weight) {
        curbWeightInPounds = weight;
    }

    this.refuel = function() {
        return "Refueling Vehicle with regular 87 octane gasoline";
    }

    this.mainTasks = function() {
        return "Driving to work, school, and the grocery store";
    }
}
```