**Metotlar İle Kapsülleme** Alanları gizlemenin yollarından bir tanesi de metotları kullanmaktır. Bir özelliği bir deyimde kullandığınızda onu ya bir okuma bağlamında (değerini değiştirmediğiniz durumlar) ya da bir yazma bağlamında (değerini değiştirdiğiniz durumlar) kullanırsınız.

**ARAYÜZ ÖZELLİKLERİ** Arayüzler yöntemlerin yanında özellikler de belirtebilir. Bunu yapmak için get ve/veya set anahtar sözcüğünü kullanırsınız. Ancak ara yüzler uygulama kodları barındıramayacakları için get ve set erişimcilerinin gövdeleri yerine noktalı virgül (;) kullanılır.

**Otomatik Özellikler:** Özelliklerin başlıca amacı alanların kullanımını uygulamasını dış dünyadan saklamaktır. Özelliklerin amacına yönelik kullanımı önemlidir. Çünkü get ve set erişimciler, sadece alana bir değer atamak ya da değer okumak işlemlerini yerine getiriyorsa bunun doğru bir yaklaşım olup olmadığı sorusu akla gelebilir. Veriyi ortak alanlar olarak göstermek yerine özellikler tanımlamanız için en az iki neden vardır:

**Arayüzlerle Uygunluk:** Bir ara yüz gerçekleştiriyorsanız ve ara yüz özellik olarak bir öğe tanımlıyorsa özellik sadece private alanlardaki veriyi okuyor ve yazıyor olsa bile ara yüzdeki şartlara uyan bir özellik yazmanız gerekir. Sadece aynı adla bir ortak alan göstermek yoluyla bir özellik geliştiremezsiniz.

**Uygulamalarla Uygunluk:** Alanlar ve özellikler, derlemelerde farklı meta veri kullanarak kendilerini ortaya koyar. Bir sınıf geliştiriyorsanız ve ortak alanlar kullanmaya karar verdiyseniz bu sınıfı kullanan bir uygulama, alanlar olarak bu öğelere başvuracaktır. Bir özelliği okurken ve yazarken kullandığınız söz dizimin bir alanı okumak ve yazmak için kullanabilseniz de derlenen kod gerçekte çok farklıdır. Daha sonra bu alanları özellikler olarak değiştirmeye karar verirseniz mevcut uygulamalar yeniden derlenmeden sınıfın güncellenmiş sürümünü kullanamaz. Uygulamayı bir şirketin çok sayıda kullanıcısına uyguladıysanız bu bir dezavantajdır.