this

In the last exercise we assigned the `area` field in the constructor:

```
class Forest
{
  public int Area
  { /* property omitted */ }

  public Forest(int area)
  {
    Area = area;
  }
}
```

The parameter for the constructor `area` looks a lot like the old field `area` and the new property `Area`. It's good to be explicit when writing code so that there is no room for misinterpretation. We can refer to the current instance of a class with the `this` keyword.

```
class Forest
{
  public int Area
  { /* property omitted */ }

  public Forest(int area)
  {
    this.Area = area;
  }
}
```

`this.Area = area` means "when this constructor is used to make a new instance, use the argument `area` to set the value of this new instance's `Area` field".

We would call it the same way:

```
Forest f = new Forest(400);
```

`f.Area` now equals 400.

The word `this` might seem frustratingly vague. Think back to the "class is to instance as blueprint is to house" analogy. The class/blueprint has to use the generic `this` because the class/blueprint is going to be reused for every instance/house.

☑**Instructions**

1.

Specify the instance properties by using `this.Name` and `this.Biome`.

Hint

Replace

```
Name = name;
Biome = biome;
```

with

```
this.Name = name;
this.Biome = biome;
```