Variables

What we will learn:

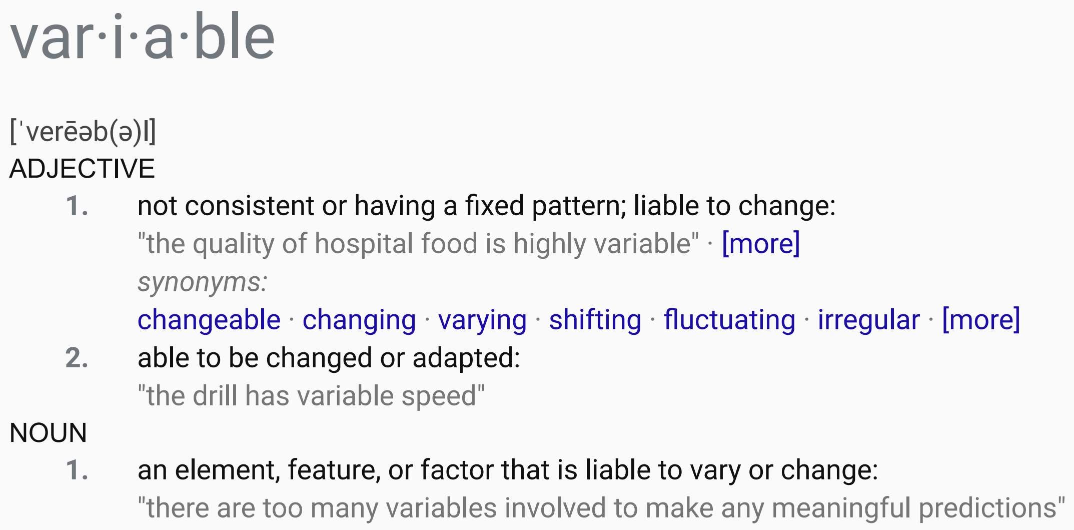
- What a variable is
- What a variable can store
- How to declare a variable
- How to initialize a variable

Why do we need to know?

- We need them to write flexible programs
- Use variables to represent data

$$x + y = ?$$

What is a variable?



var·i·a·ble

[ˈverēəb(ə)l]

ADJECTIVE

1. not consistent or having a fixed pattern; liable to change:
   "the quality of hospital food is highly variable" · [more]
   synonyms:
   changeable · changing · varying · shifting · fluctuating · irregular · [more]
2. able to be changed or adapted:
   "the drill has variable speed"

NOUN

1. an element, feature, or factor that is liable to vary or change:
   "there are too many variables involved to make any meaningful predictions"

What is a variable in programming?

- A **container** that stores some value
- Variables keep track of the data throughout the program
- Variables are used to store, retrieve, and modify data

Basket 1
Basket 2
Basket 3
4
16
75

Opposite of a variable

- A Constant
- *Constants are values that never change.*

```
const int MonthsInAYear = 12; //can’t change
```

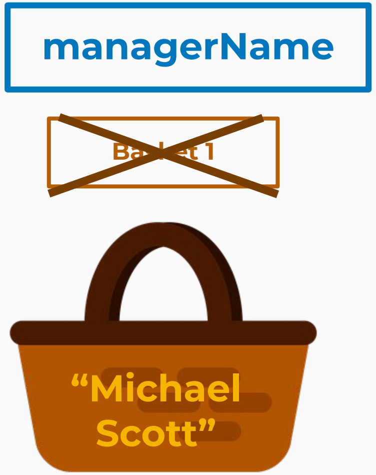
What is a variable?

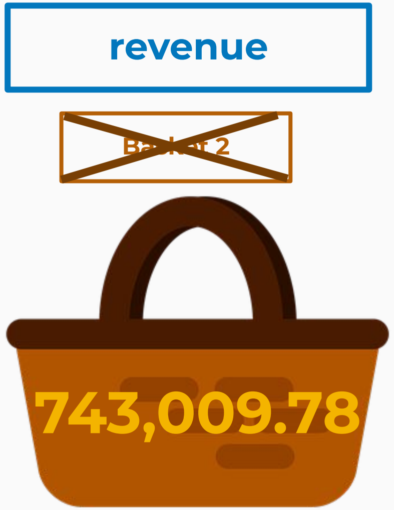
– values that *CAN* change!

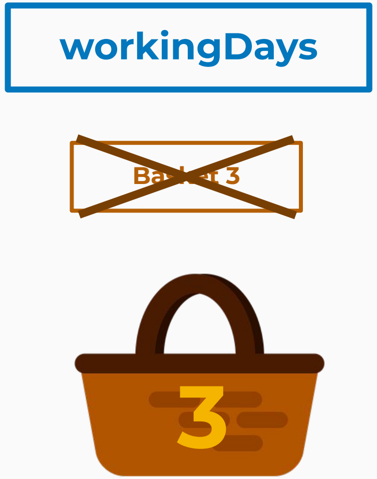
```
int numberOfApples = 12; //can change
```

Things to consider:

1. Name of your basket
2. Size of your basket
3. What type can it hold?







C# is Strongly and Statically Typed

1. Strongly - Once a variable has a type, that type cannot change
2. Statically - A variable MUST have a type

string managerName =

"Michael Scott"

double revenue =

int workingDays =

C# Syntax

```
datatype variableName;  // Declaration

variableName = value;  // Initialization
```

C# Syntax

```
datatype variableName;   // Declaration

variableName = value;   // Initialization



datatype variableName = value;   // Declaration &
                                  // Initialization Syntax
```

C# Syntax

```
datatype variableName;   // Declaration

variableName = value;   // Initialization



datatype variableName = value;   // Declaration &
                                  // Initialization Syntax



 int numberOfApples;      // First Example
 numberOfApples = 12;
```

🍎

C# Syntax

```
datatype variableName;   // Declaration

variableName = value;   // Initialization



datatype variableName = value;   // Declaration &
                                 // Initialization Syntax
```

🍎
```
int numberOfApples;     // First Example
numberOfApples = 12;
```

🍐
```
int numberOfPears = 12;        // Second Example
```

C# Syntax Do’s

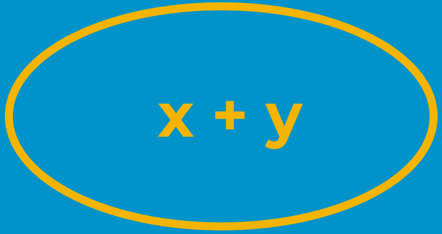
```
Yes:
  ●   camelCase for variables
  ●   Descriptive name
  ●   Can contain letters, numbers, and
      underscore – that’s it
```

```
string managerName = “Michael Scott”;

double revenue = 743,009.78;

int workingDays = 3;
```

C# Syntax Do’s



Yes:

- camelCase for variables
- Descriptive name
- Can contain letters, numbers, and underscore – that’s it

```
string managerName = “Michael Scott”;

double revenue = 743,009.78;

int workingDays = 3;
```

C# Syntax Do’s

revenue + expenses

Yes:

- camelCase for variables
- Descriptive name
- Can contain letters, numbers, and underscore – that’s it

```
string managerName = “Michael Scott”;

double revenue = 743,009.78;

int workingDays = 3;
```

C# Syntax Don'ts

No:

- Cannot have spaces
- Cannot start with a number
- Cannot start with a symbol
- Cannot be a reserved keyword like string, return, if, etc.
- Cannot start with a dash

INVALID:

```
string Manager Name = “Michael Scott”;

string 4managerName = “Michael Scott”;

string ~managername = “Michael Scott”;

string string = “Michael Scott”;

string -managerName = “Michael Scott”;
```

String Interpolation

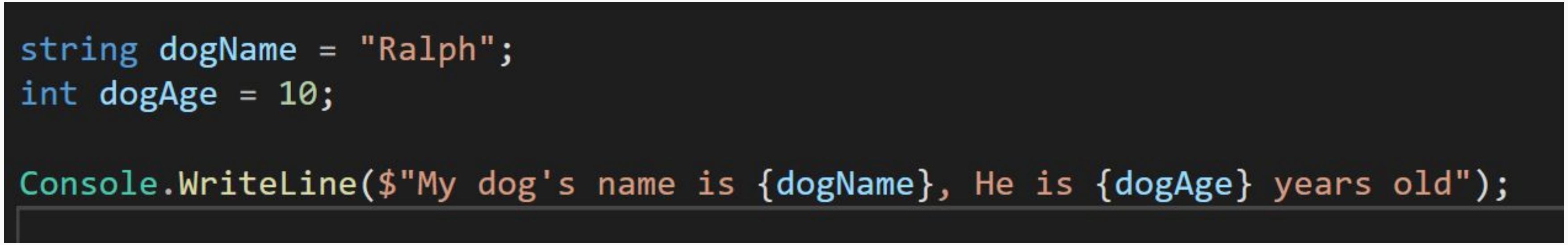
```
string dogName = "Ralph";

int dogAge = 10;



Console.WriteLine($"My dog's name is {dogName}, He is {dogAge} years old");
```

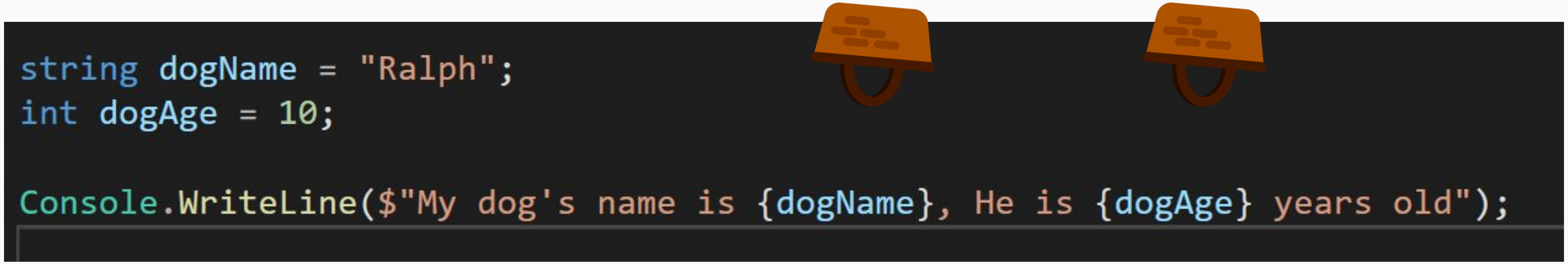
String Interpolation



```
string dogName = "Ralph";
int dogAge = 10;

Console.WriteLine($"My dog's name is {dogName}, He is {dogAge} years old");
```

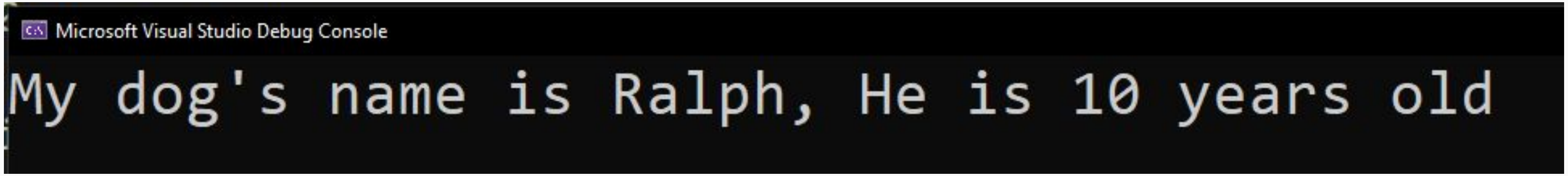
String Interpolation



```
string dogName = "Ralph";
int dogAge = 10;

Console.WriteLine($"My dog's name is {dogName}, He is {dogAge} years old");
```

Output



Microsoft Visual Studio Debug Console

```
My dog's name is Ralph, He is 10 years old
```

Takeaways

1. Variables act like containers
2. Variables allow us to store, retrieve, and modify data
3. C# is strongly and statically typed

Variable Exercise Bonus:

Research Console.ReadLine() and implement it in your exercise

Variables Demo

Declaring and Initializing Variables

C# is Strongly and Statically Typed

Constant