Enums and strings

CONTENTS

- **Objectives**
  - Explore enums and strings

- **Contents**
  - Enumerated types - keyword `enum`
  - `string` and `StringBuilder`
  - Review

- **Hands-on labs**

Enumerated data types

Using the 'class' keyword is not the only way to define a type

- A variable of the `enum` type is restricted to a set of predefined constants
- Treat `enum` variables just like other value types

```
public enum Status {
    Active,
    Retired,
    Contractor,
    Permanent
}
```

```
Status status = Status.Retired;
ExpectStatus(status);
```

```
public void ExpectStatus (Status status)
{
    if (status == Status.Active)
        // ... process Active

    else if (status == Status.Retired)
        // ... process Retired

}
```

Strings

Strings are immutable

```
string name = "Bob";
string name = new String("Bob");
```

Is like

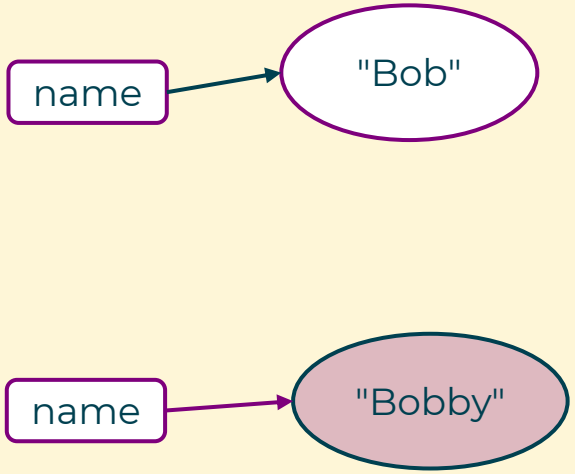
```
name[0] ='R';
```

```
name = name + "by";
name = new String(name + "by");
```

Is like


```
public static void Main(string[] args) {
    String name = "Bob";
    ChangeName(name);
}

private static void ChangeName(string name) {
    name = "Bobby";
}
```



String methods

String has a method `Substring`. What will this code display?

```
string s1 = “Shampoo";
Console.WriteLine(s1.Substring(1, 3));
```

- and this?

```
string s1 = "Fred";
s1 += " Bloggs    ";
S1 = s1.Trim();
s1 = s1.Substring(0, 5).ToLower() + s1.Substring(5, 6).ToUpper();
Console.WriteLine(s1);
```

C#: Other String methods

```
string s = "Fred Smith";
char c = s[2];              // would be ‘e’
foreach (char c in s) {..}

s.EndsWith("h");       // returns a Boolean value
s.StartsWith("F");

int i = s.IndexOf('r');
string[] words = s.Split(" ");

s.Length;
s = s.Replace("ed","eddy");
s = s.Trim();
```

- None of these methods are `void`, you need to catch what they return

StringBuilder

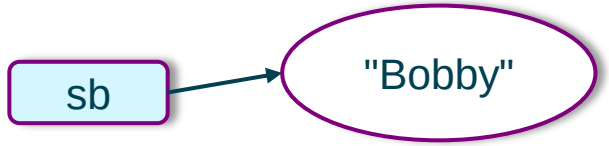
StringBuilder is a mutable String buffer

- Key methods: `Append()`, `Insert()`, `Replace()`, `Remove()`

```
StringBuilder sb = new StringBuilder("Bob");

sb.Append("by");
```

```
string name = sb.ToString();
```



StringBuilder class - Examples

```
string s = "Fred";
for (int i = 0; i < 200; i++) {
  s += "a";
}
Console.WriteLine(s);
```

It would display ‘Fred’ followed by 200 ‘a’s

But generates 200 string objects to be garbage collected

- **Any change to a String creates a new String.** Old one will be garbage collected
  - StringBuilder uses a buffer space in memory

```
StringBuilder sb = new StringBuilder("Fred");
for(int i = 0; i < 200; i++) {
      sb.Append("a");
}

Console.WriteLine(sb.ToString());
```

```
Fredaaaaaaaaaa.....
```

Review

- **Enumerated types - keyword enum**
- **classes String and StringBuilder**
  - `String` is a class but with many value type behaviours
  - `StringBuilder` – mutable string buffer with Append, Replace, Insert…

Hands-on Lab

- **Part 1 –** Defining and using `enum`
- **Part 2 –** Using `String` and its key methods
  - Also using class `StringBuilder`