Objects and Components

Rob Miles

Historical Relic....

The World's First Visual Basic Programmer....

Objects and Components

Rob Miles

Department of Computer Science

0708 07 08120 Programming 2

THE UNIVERSITY OF HULL

Welcome

- Welcome to the lecture
  I hope you soon be hearing
- A whole new bunch of wondrous facts
  About Software Engineering
- And if for some strange reason
  You do not find it funny
- That need not pose a problem
  I've already got your money…

Learning Outcomes

- I can't make many promises
  About what you will find out
- A bit on C# coding
  and what objects are about
- But even if you concentrate
  Apply your brain cells many
- I'm afraid it is unlikely
  You'll find out where's the key marked 'any'

Software Engineering

- So what is all this software stuff?
  And why is learning it so tough
- In this bit we will find out more
  Get a handle on the software score

Software

- Now software it is funny stuff
  You can't hit it with a hammer
- And sometimes it does misbehave
  I blame the Microsoft programmer
- But people seem to live with
  Software that does jar
- Do you recall the last time
  You had to reboot your car?

Engineer

- If you thought that engineers
  Were simply folks with spanners
- Who are lacking in their dress sense
  And have appalling manners
- Then stick around and learn stuff
  There is no need to fear
- Dear audience I bring you
  The Software Engineer!

Software Engineer

- Software Engineers do cunning stuff
  All high level design
- And if the job goes down the tubes
  It's them that should resign
- But if you want to upset one
  And seen them spit and stammer
- Just refer to them in public
  As but a mere "programmer"

Engineering Software

- Now working hard with metal
  Is easy to relate
- And just what makes a bridge good
  Is not open to debate
- But software it is funny stuff
  It's not physical moreover
- It is sometimes hard to tell just when
  Your software will fall over

Vista Roll over and die

kernel fault at F4C0:04C44

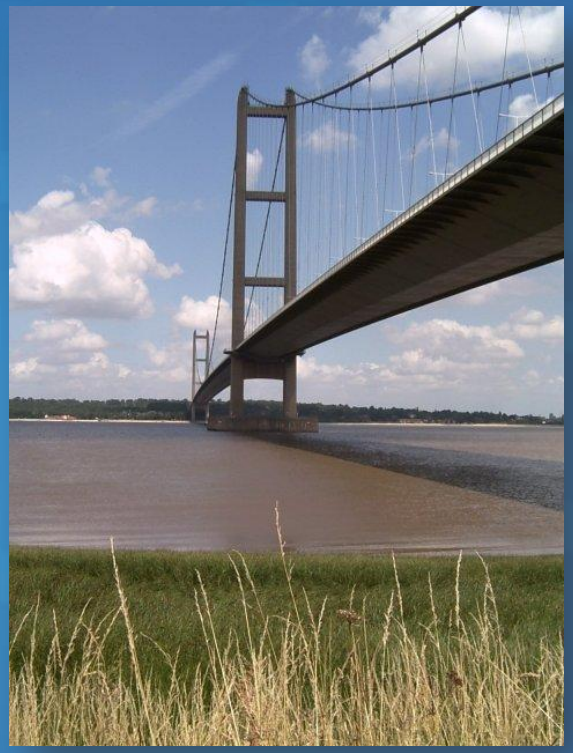
1231231A 11FC3005 23221230 12320DC0 1232105C 23423FC0
0000000B 00000000 0000000D 41424344 45464748 494A4B4C
1231231C 11FC3005 23221230 12320DC0 45464748 494A4B4C
3221230D 12320DC0 01231231 11FC3005 23221230 12320DC0
1231231E 11FC3005 12320DC0 01231231 11FC3005 23221230
1231231F 11FC3005 12320DC0 01231231 11FC3005 23221230
4546474G 1494A4B4 45464748 494A4B4C 12312310 11FC3005
12312310 11FC3005 12320DC0 01231231 11FC3005 23221230
45464748 494A4B46 45464748 494A4B4C 1231231 11FC30054
45464748 494A4B47 45464748 494A4B4C 1231231 11FC30054
45464748 494A4B48 45464748 494A4B4C 1231231 11FC30054
45464748 494A4B49 45464748 494A4B4C 1231231 11FC30054
45464748 494A4B40 45464748 494A4B4C 1231231 11FC30054
45464748 494A4B41 45464748 494A4B4C 1231231 11FC30054

AX :12312343
BX :45645666
DX :3412312D
EX :12312343

System dump in progress

Software & Hardware

- When you build a bridge
  You can make it using steel
- You know about the parts
  And how they'll work for real
- But working with the software
  Is a much more taxing thing
- You never can be quite sure
  What errors it will bring



Software Components

- To make our software stronger
  And get the best effects
- The sharper person writes
  Their code using objects
- Objects are just lumps of code
  That can be put together
- It seems that writing code this way
  Is really rather clever

Objects

- People like objects
  They know them quite well
- Objects have properties
  Like colour, taste and smell
- Some objects do things
  And useful they are too
- Once you have got an object
  You can make it work for you

Software Objects

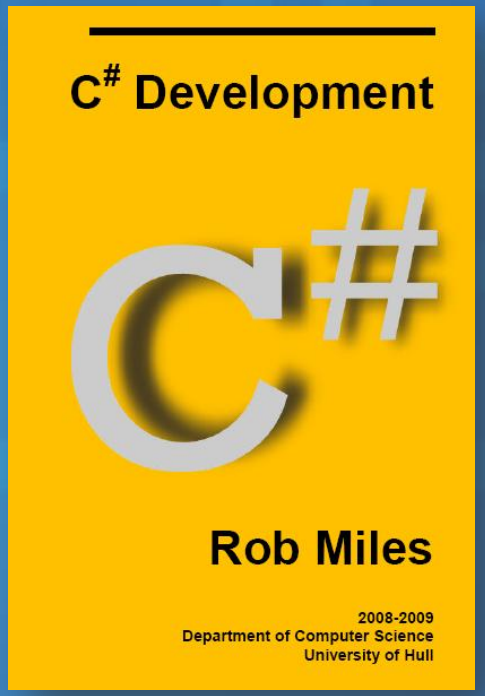
- An object is a melded lump
  A data and code compaction
- The data lets it store stuff
  The code does all the actions
- The things inside the object
  You would do well to remember
- Are usually described as
  The objects members

Why an Object?

- If you put code in with the object
  The good thing that this brings
- Is that it stops naughty programmers
  From doing dodgy things
- The member data is protected
  And the thing that you can see
- Is that all your clever methods
  Can ensure integrity

Objects & C#

- Objects they are really cool
  All knowledgeable and smart
- If you want to write good ones
  C#'s the place to start
- C# is a language
  'tells the system what to do
- Just create some C# classes
  And they'll do the work for you



Sample Object

```
class Poem
{
    private string poemText;
}
```

- Here is a sample class
  in C# it is written
- It contains a string data member
  For poem storage it is permitting

Private Parts

```
class Poem
{
    private string poemText;
}
```

- The **poemText** is private
  To hide it from the world
- You don't want some one changing it
  For some words that don't rhyme

Public Access

```
class Poem
{
    private string poemText;

    public void Tell ()
    {
        Console.WriteLine(poemText);
    }
}
```

- To get our poem to the outside world we use the method **`Tell`**
- This will print our poem words onto the conso-el

Storing Our Words

```
class Poem
{
    private string poemText;

    public void Store ( string newText )
    {
        poemText = newText ;
    }
}
```

- To put the words inside our poem we use the method **`Store`**
- This places our new masterpiece on what was there before

Constructing a Poem

```
Poem ode = new Poem();

ode.Store("The poem goes here" +
     "and will not disappear.");

ode.Tell();
```

- The C# you see
  Stores some poetry for me
- The **`Tell`** method call
  Shows it on the cons-aul

Encapsulation

- What we have been showing
  Is a programming creation
- Designed to show the principle
  Of encapsulation
- You hide members in the objects
  To save them from corruption
- And then access them from methods
  This is object based construction

Inheritance

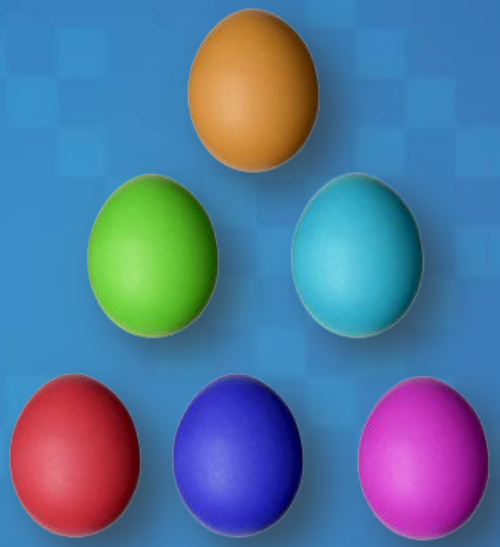
- Encapsulation's one thing
  That makes us like our objects
- The object receives messages
  And bad requests it rejects
- Inheritance is also useful
  As an object skill
- And it is nothing that's to do with
  A program that writes it's will

Inheriting

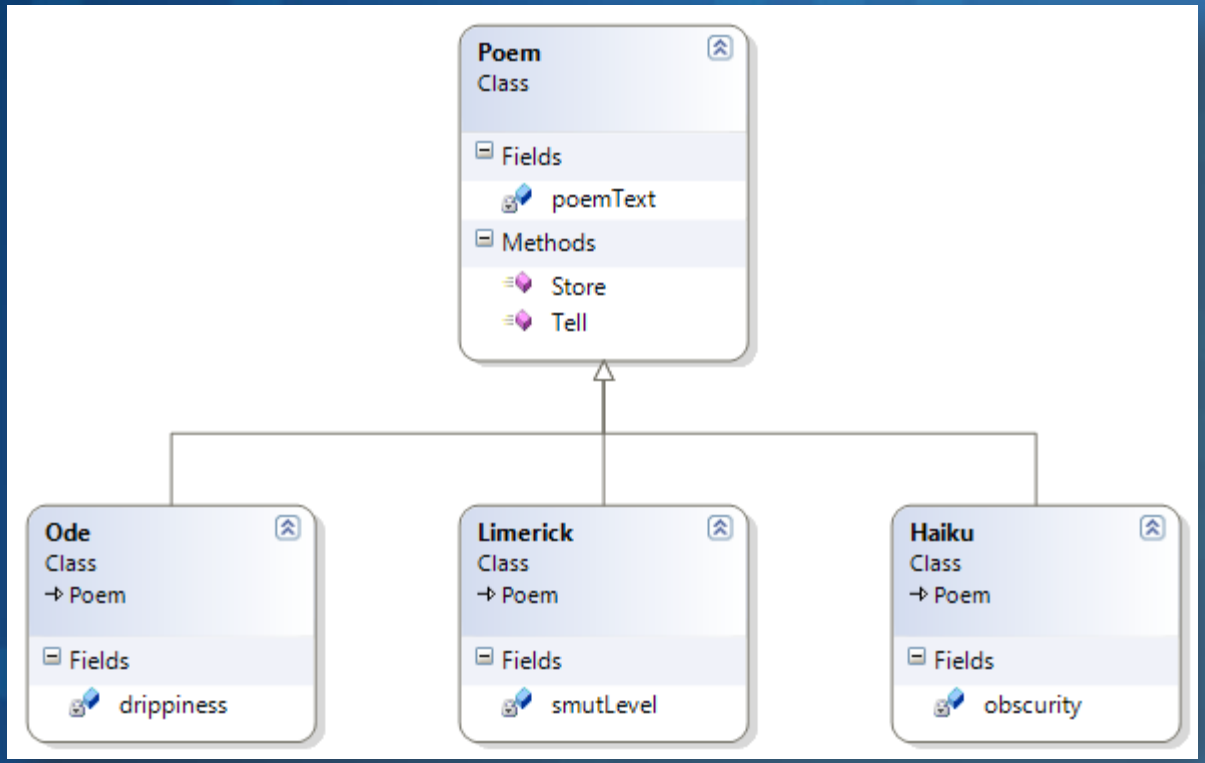
- Inheriting in this case means
  The stuff from mum and dad
- Picking up the good stuff
  Along with all the bad
- When working out inheritance
  It says in all the books
- Make sure you get your fathers brains
  And your mothers looks

Objects & Inheritance

- Inheritance with objects means
  You build a family tree
- So classes can be based on
  What they should nearly be
- We can now make new classes
  A hierarchy in fact
- All springing from a base class
  The start of all the act-tion



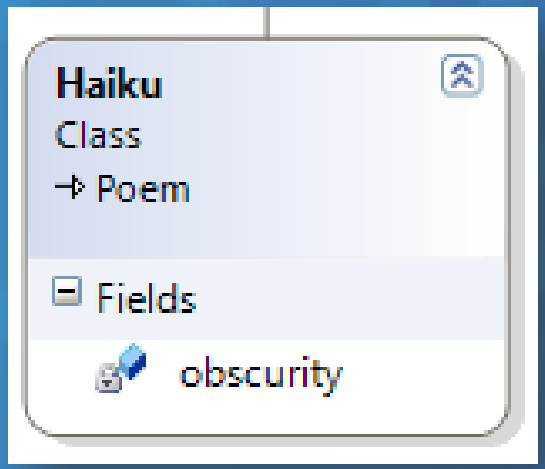
Poem Inheritance



- Above you can see
  The Poem Family Tree

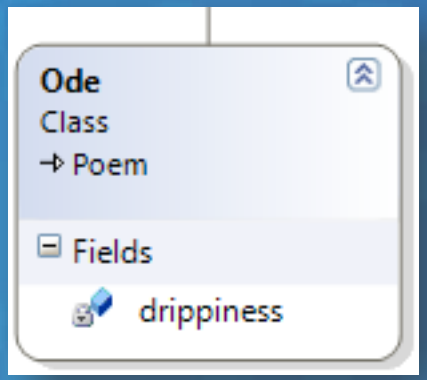
Haiku

- Haiku is a 17-syllable verse form consisting of three metrical units of 5, 7, and 5 syllables:
  - Haiku is tricky
    Getting the counts right is hard
    I got it wrong
- Haiku has a quality
  that I call obscurity



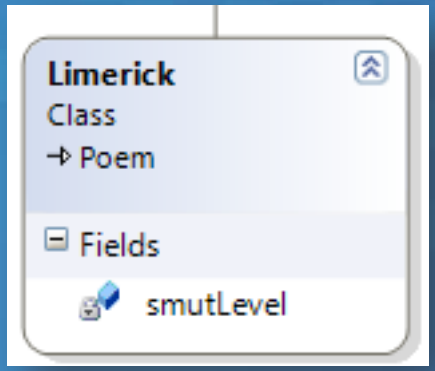
Odes

- Odes are like what Keats wrote
  To earthenware in Greece
- I think they are rubbish
  And I've made a property
- It indicates their drippiness
  The higher the more drippy
- The amount of drugs that Keats took
  I should really call them trippy



Limerick

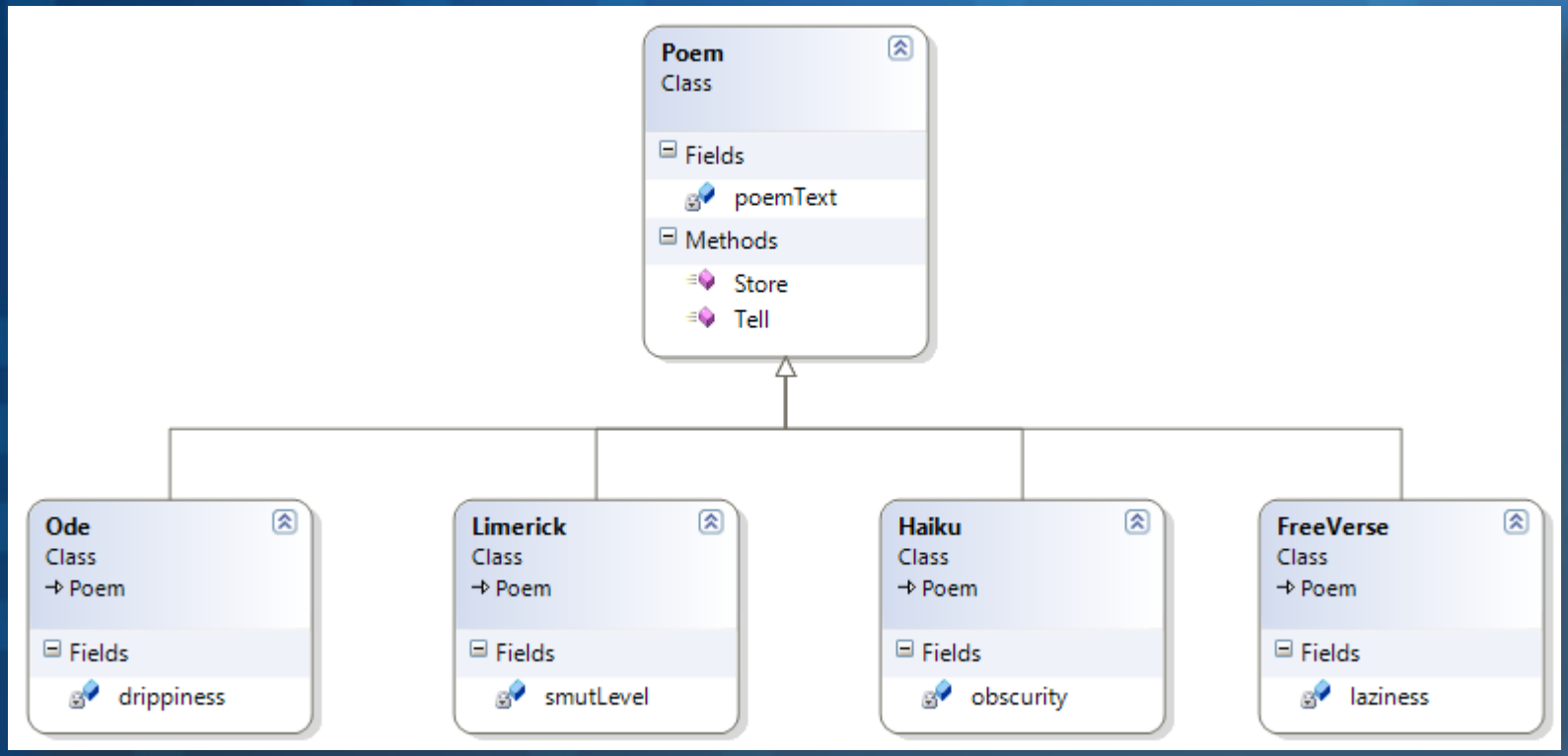
- There once was a fellow called Miles,
  Who at rhyming had several styles,
  He once tried a limerick,
  But it turned out disastrous,
  And he had to revert back to prose
- Limericks are five line poems
  Which are sometimes rather shady
- Ones which have smutLevel 10
  Start "There once was a lady…"



Using Inheritance

- If I think of a new type
  For example there is free verse
- I simply add a new class
  As down the tree I traverse
- The new class can do everything
  The parent is providing
- And you can change existing behaviours
  With a bit of overriding

The New Tree


Poem
Class
Fields
poemText
Methods
Store
Tell
Ode
Class
Poem
Fields
drippiness
Limerick
Class
Poem
Fields
smutLevel
Haiku
Class
Poem
Fields
obscurity
FreeVerse
Class
Poem
Fields
laziness

Virtual Reality

```
class Poem
{
    private string poemText;

    public virtual void Tell ()
    {
        Console.WriteLine(poemText);
    }
}
```

- The method in the parent
  We might want to replace
- Must now be marked as virtual
  So overriding can take pace

Overriding Importance

```
public class Limerick : Poem
{
    private int smutLevel;
    public override void Tell()
    {
        if (smutLevel > 5)
        {
            removeChildren();
        }
        base.Tell();
    }
}
```

- Limericks can be so smutty
  We don't want kids to hear
- We override the Tell method
  To keep us in the clear

Overriding Importance

```
public class Limerick : Poem
{
    private int smutLevel;
    public override void Tell()
    {
        if (smutLevel > 5)
        {
            removeChildren();
        }
        base.Tell();
    }
}
```

- When we've got the kids out
  The thing that we can do
- Is use the base behaviour
  To tell the thing to you

Inheritance Fun

- So inheritance, it lets us use
  Objects we made before
- Although we will not let it stop
  Us charging customers more
- With objects and good class design
  Great flexibility is what we get
- But we haven't reached the very end
  Of what objects can do yet

Polymorphic Poems

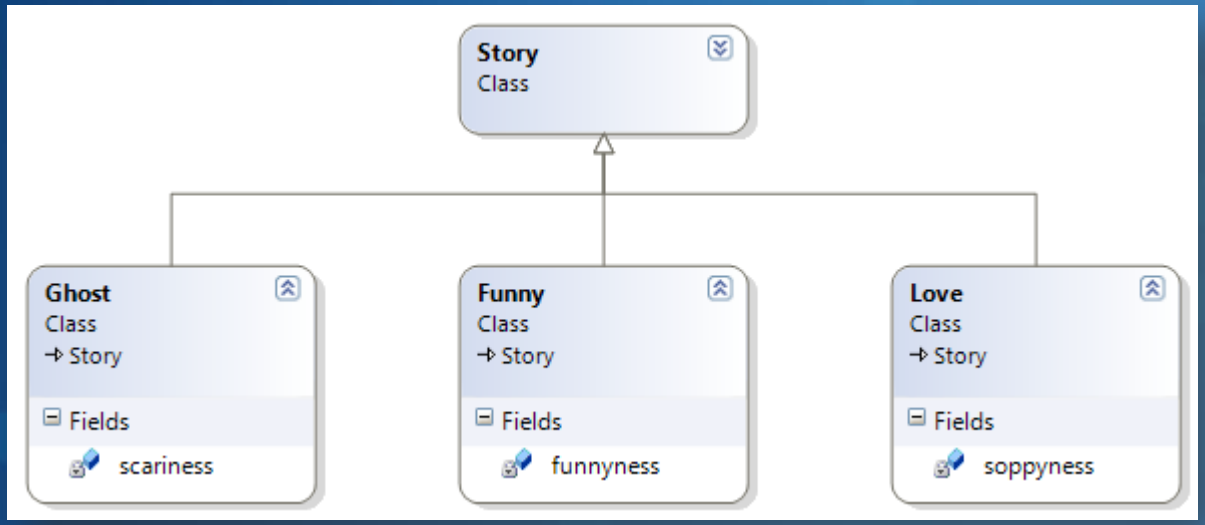
- Polymorphism is the last trick
  And very hard to rhyme with
- A rhyming dictionary I
  Could go and spend some time with
- Another idea comes to me
  Could this lecture be much worse
- I could always do the rest of it
  In a cop out known as "free verse"

But I won't!

Polymorphism

- Polymorphism lets us think about
  The things that objects do
- The action of our Poems
  Is to tell themselves to you
- But there may be other objects
  That we may wish to see
- That also wish to tell themselves
  For example, there's a Story

Story Time



- The Story objects, it would seem
  Look just a bit like the po-eem

Tell Me Something

- When thinking of objects
  We have many different kinds
- The Story it has narrative
  the Poem it has rhymes
- But when you think how to use them
  It might be just as well
- To think of these two different types
  Having the action "`Tell`"

The telling Interface

- Both classes have a method
  Which when called will make them tell
- Perhaps I could add a joke class
  Which could have one as well
- When considering what the classes do
  In C# the proper place
- Is to put the actions to be done
  Inside an interface

Interfaces

```
public interface Ientertainer
{
    void Tell();
}
```

- This is the interface
  That gives our objects voice
- It lets us ask them to tell themselves
  In a manner of their choice

Implementation

```
class Poem : IEntertainer
{
    private string poemText;

    public void Tell ()
    {
        Console.WriteLine(poemText);
    }
}
```

- We say our Poem can tell
  And get the message out
- By IMPLEMENTING INTERFACES
  (but there's no need to shout)

Renaissance Classes

- An class which can do many things
  “Renaissance” if you like
- Like read a poem, print it out
  and even ride a bike
- Will show that it is clever
  And with luck the world impress
- By implementing many interfaces
  To show off it’s success

Poem Power

```
class Poem : ITell, IPrint, IVideo
{
    // A very clever object
    // works with interfaces plenty
    // There is no upper limit
    // tho' I'd draw the line at twenty
    // But for every interface you use
    // to keep the compiler sane
    // the methods must be present
    // which the interfaces contain
}
```

Polymorphic Poems

- The **Poem** is polymorphous
  A versatile beast indeed
- Ask a **Poem** to tell itself
  And your request it will succeed
- To a printer it is different
  With printing it is good
- To each user the **Poem** presents the face
  That user thinks it should

Polymorphic Power

- Polymorphism makes objects show off many faces
- The same object can be utilised In lots of different places
- In every single context And this is the clever bit
- The interface that it is used will always make it fit

And finally…

- Community Singing…

Get by with a little help from C#

What would you say if I told of something
that would help you get your software right?
Hand me your cash and I'll tell you some more
And then I'll run off out of sight

Oh I get by with a little bit of C#,
Mmm I design with a little bit of C#,
Mmm I must try with a little bit of C#

Click to add title

What do you do with frown on your face?

And a program that you've got to write

Just open up Studio and get on the case

You can build all the Forms in one night

Oh I get by with a little bit of C#,

Mmm I design with a little bit of C#,

Mmm I must try with a little bit of C#

You can add a title by clicking here

Do you need a new language?

- C# is the place to get more

Could it be a new language?

- It will be when we get Version 4

Click here to add a different title

What do you do when it needs to be right?

And the project it is running out of time

What do you do when you don't want to fight

With constructions that seem to be a crime

Oh I get by with a little bit of C#,

Mmm I design with a little bit of C#,

Mmm I must try with a little bit of C#

Does nobody like titles any more?

(would you believe that this time it's for real)
That now software we can really engineer
What makes the programs with greatest appeal
And leaves the customer with nothing to fear

Oh I get by with a little bit of C#,
Mmm I design with a little bit of C#,
Mmm I must try with a little bit of C#

THANK YOU.. AND GOODBYE…

NO MONEY REFUNDED!