AS Computer Studies: PROGRAMMING

WRITING TO TEXT FILES

Starter

- You are about to see a picture on screen.
- The picture will be displayed for 45 seconds.
- After the 45 seconds you will be given a piece of paper and you should try and re-draw what you have seen.

Alternative:

- Two teams
- One member of each team, 10 seconds at a time, will look at the picture.
- Then they will return in their teams and write / draw on paper what they've seen.

I II III IV V VI VII VIII IX X

10101010

ICT
ENGLISH
MATHS
SCIENCE
HISTORY
GEOGRAPHY
PE
TECHNOLOGY
ART
RE
ENRICHMENT

$X = 5$

$Y = (X*X)+2 = 27$

Starter Review

- You just used a similar **concept of programming** in this starter.
- You looked at something, **created a space in your memory** to remember parts.
- Yes, the computer is more accurate over time, but you both have one thing in common.
- If you don't save (write down), you will forget a lot of what you have just seen.
  - Hence, me nagging you to take notes!!!

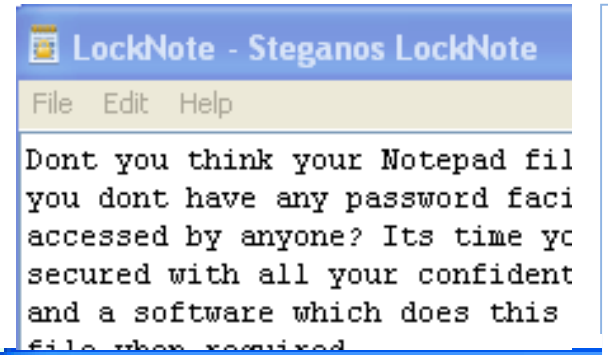
Objectives

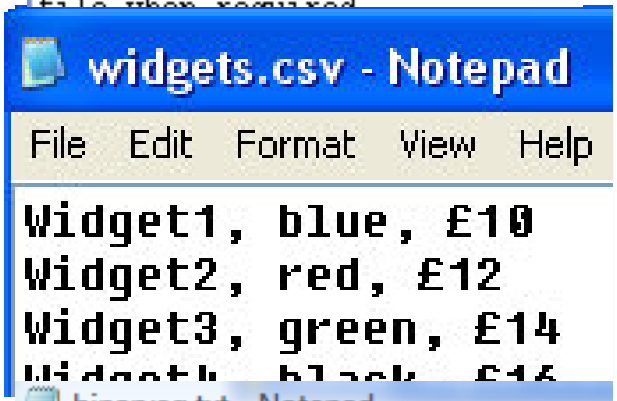
- Understand the different ways in which a computer can **read** and **write** data **to file**.
- Become familiar with the coding constructs of **saving and reading data**.
- Use saving and reading within your program.

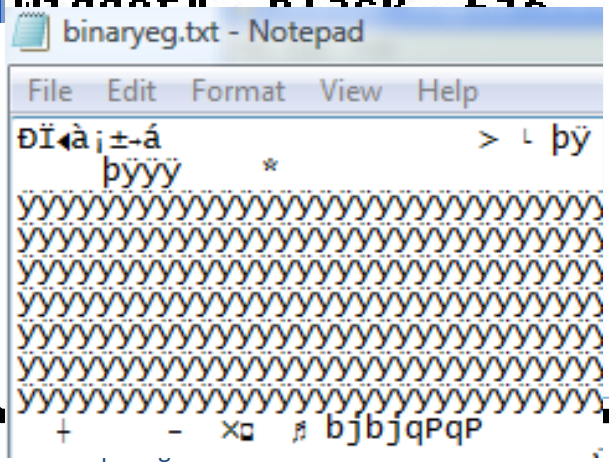
The Concept

- So far, you have been writing programs that create and use variables and constants.
- These are in **RUNTIME memory**.
- When the program terminates, the contents of the variables are lost. In other words, the computer forgets!
- Imagine if it did this to a word processing document!
- To get around this, **we can write our data to disk**, and read it back when needed.
- There are a few ways we can do this.

File Types







- **Text File (extension .txt).** Basically a set of character values which are in a document line by line. You could open it in Notepad or similar application and read what was inside it.
- **CSV File (extension: .csv).** A comma-separated values file. Like a text file, with all related information on one line (e.g. all information about a customer: name, age, debt etc), with **the fields separated by commas**.
- **Binary file – (extension varies).** A application, which is opened in say notepad would not be comprehensible to the human eye.

Preparing to Read / Write

- Your program needs instructions from a **library**.
- It needs to know **HOW to read/write to files**.
- At the top of your module write:

Imports System.IO

Writing Console Entries to File – the StreamWriter method

1. Add "**Imports System.IO**" to your General section.

2. Declare a **channel for writing** (add writing functionality using StreamWriter

Dim fileWriter as StreamWriter

3. Specify the **file you want to write to** (in a variable or constant – useful to put at top of sub).

```
Dim filename as string
fileName = "c:\writerExample.txt
```

4. **Bind the channel to the file** (i.e. tell the computer what file you are going to write to, using your writing channel.

fileWriter = New streamwriter(fileName)

Writing Console Entries to File – the StreamWriter method

5. Prompt the user to **enter something in the console** using the writeline command. Then **store** what the user types in a variable. E.g. **strEntry = console.readline()**

6. **Write** what was written in the console **to file**
fileWriter.writeline(strEntry)

7. **Close the file**.
fileWriter.close

Let's look at a coded example (handout).

Today's Task 1

Writing to file

- **FIRST**:
  - Attempt questions 1 and 2 from Task 16 in the pink booklet.
  - Hint: For task 2 you will need to allow the user to type a keyword to stop the loop (e.g. End).
  - Don't forget to actually CHECK the text file once you have written to it.
- **THEN:**
  - You could go back and open ANY program you have written and adapt it so that it writes the result to file, as well as displaying it on-screen. (This is worth doing for practice).

Reading from a File – the StreamReader method

1. • Add "**Imports System.IO**" to your General section.

2. • Declare a channel for reading(add writing functionality using StreamReader
   Dim fileReader as StreamReader

3. • Specify the file you want to read from (in a variable or constant – useful to put at top of sub).
   Dim filename as string **fileName = "c:\writerExample.txt**

4. • Bind the channel to the file (i.e. tell the computer what file you are going to write to, using your writing channel. **fileReader = New streamReader(fileName)**

5. • Read each line from the file, until the file is empty. (**Do Until FileReader.EndofStream)…Loop**.
   • Use **FileReader.Readline()** within the loop to read a line. e.g. **strEntry = filereader.readline()**

6. • Display the contents of the variable on screen: **console.writeline(strEntry)**

7. • Close the file. **fileReader.close**

Today's Task 2

- Time to rehearse reading from file.
- **FIRST**:
  - Attempt question 3.
- **THEN:**
  - Write a program which takes the file from Task 2 and displays the contents on screen.

Objectives

- Understand the different ways in which a computer can **read** and **write** data **to file**.
- Become familiar with the coding constructs of **saving and reading data**.
- Use saving and reading within your program.

Plenary: What is the correct order?

A. Add "**Imports System.IO**" to your General section.

B. Bind the channel to the file (i.e. tell the computer what file you are going to write to, using your writing channel. **fileWriter = New streamwriter(fileName)**

C. Prompt the user to enter something in the console using the writeline command
- Store what the user types in a variable. E.g. **strEntry = console.readline()**

D. Declare a channel for writing (add writing functionality using StreamWriter
Dim fileWriter as StreamWriter

E. Write what was written in the console to file **fileWriter.writeline(strEntry)**

F. Close the file. **fileWriter.close**

G. Specify the file you want to write to (in a variable or constant – useful to put at top of sub).
Dim filename as string **fileName = "c:\writerExample.txt**