Writing Data to and Retrieving Data from a Random Access file

A random access file stores data as ASCII text in a structured format. These random access files contain records that can be accessed in any sequence. This is in contrast to a sequential text file in which data much be accessed from the beginning. Other differences in a random access file versus a sequential text file include

- Sequential - Must be opened specifically for writing or reading
- Random - Once opened, data may be either written or read

- Sequential - Ends each file (EOF) with the line feed Chr$(13);Chr$(10)
- Random - No End of File (EOF) line feeds are added

- Sequential - Supports comma separated fields
- Random - Fields are defined by a specific number of characters

- Sequential - Can be easily and sensibly viewed with any text editor
- Random - Can be opened with a text editor, but not easily viewed

- Sequential - Entire contents must be rewritten when file is modified
- Random - A single record can be rewritten when the file is modified

Because there are no End of File (EOF) markers, any random access file opened as a text file will appear as one single line of text.

Random Access File Commands and Operations

There are three basic operations in random access files.

1. Creating the random access file
2. Writing each record of the random access file
3. Reading each record of the random access file

Basic Random Access File (RAF) commands and functions include

- OPEN - used to access the file
- LEN - the number of ASCII characters allotted for reach record
- FIELD - used to define the fields, or categories, of the records
- PUT - used to write a record to the file
- GET - used to retrieve a record from the file
- GETTRIM - used to trim padding from fields in the retrieved record
- LOF() - the size of, or number of ASCII characters in, the entire file

Creating a Random Access File

A random access file requires some thought before creation. Questions to ask are

- How many fields are needed?
- What is the maximum number of characters needed for each field?

Each record contains several fields. This tutorial will create a random access file of world leaders. Each record will contain country, title, and last name. 30 characters will be allotted to the country name, 20 characters will be allotted to the title, and 20 characters will be allotted to the last name. That's a total of 70 characters, or bytes, allotted to each record. 70 is the LEN of the record. Unique variables must be assigned to each field. Field variables throughout this tutorial will be country$, title$, lname$. Finally, the file must be given a name. Here we'll use HeadsOfState.dat. Now that the minimum requirements are met, the HeadsOfState.dat random access file can be created. The format to create the random access file follows this format

```
OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
CLOSE #1
END
```

The commands OPEN, RANDOM, and LEN are case-insensitive. "HeadsOfState.dat" can be any name and extension you choose. Sequential text files commonly use the .txt extension while random access files commonly use the .dat extension. LEN must equal the total number of ASCII characters allotted to each record, which is the same as the sum total number of bytes of all fields. The file name can be a variable. LEN cannot be a variable.

```
fName$ = "HeadsOfState.dat"
nField1 = 30
nField2 = 20
nField3 = 20
OPEN fName$ for RANDOM as #1 LEN = 70
    FIELD #1, nField1 as country$, nField2 as
 title$, nField3 as lname$
CLOSE #1
END
```

#1 is also an arbitrary name. #a, #MyTextFile, #123xyz, or any other combination of numbers and letters will work as well.

Adding a Record

Once the file has been opened, data can be written into that file. Unlike a sequential text file, the random access file requires the record number stated before data is written. This tells the file where to write that data. The field names are also the variable names of the information to be written, so it is important to assign those variables prior to writing data.

```
country$ = "Japan"
title$ = "Emperor"
lname$ = "Akihito"

OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    PUT #1, 1
CLOSE #1

END
```

Open the field with any text editor, even Liberty BASIC's mainwin will do. Japan, Emperor, and Akihito will be visible, but instead of each entry occupying one line apiece, all three remain on the same line, separated by spaces. These spaces are fillers, padding each entry with the number of spaces required to meet the number of characters allotted to each field. Japan, 5 characters in length, is followed by 25 spaces for a total of 30 bytes for the country$ field. Emperor, 7 characters in length, is followed by 13 spaces for

a total of 20 bytes for the title$ field. Finally, Akihito, 8 characters in length, is followed by 12 spaces to make up the 20 bytes allotted to the lname$ field.
It is possible to get the total number of characters in the file with the LOF() command.

```
OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    FileLength = LOF(#1)
CLOSE #1
    Print "LOF = ";FileLength
END
```

Since there is only one record in the file, the length of the file is the length of the record, or 70 characters. To add another record, assign the new data to the field name variables country$, title$, and lname$ and PUT into the next record.

```
country$ = "United States of America"
title$ = "President"
lname$ = "Obama"

OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    PUT #1, 2
CLOSE #1

END
```

The HeadsOfState.dat file now contains two records. Because each field variable name must contain the required data to be written, it is sometimes easier to use a GOSUB rather than rewriting the code each time. Add records number 3 and 4 with a GOSUB.

```
nRecord = 3
country$ = "Tajikistan"
title$ = "President"
lname$ = "Rahmon"
GOSUB [WriteRAFData]

nRecord = 4
country$ = "Saudi Arabia"
title$ = "King"
lname$ = "Abdallah"
GOSUB [WriteRAFData]

END
```

```
[WriteRAFData]
OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    PUT #1, nRecord
CLOSE #1
RETURN
```

Does HeadsOfState.dat now contain 4 records? Open the file with any text editor again. All of the data remains on the same line with fields padded with spaces to reach the allotted number of characters. OPEN the file again and check the size of the file using the LOF() command.

```
OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FileLength = LOF(#1)
CLOSE #1
    PRINT "LOF = ";FileLength
END
```

- Output

```
LOF = 280
```

If 4 records have been added, then the FileLength will indeed by 4 records times 70 characters per record, or 280 bytes because each character occupies one byte. Remember that a random access file need not have records added sequentially. Insert data into the 8th record spot, bypassing records 5, 6, and 7.

```
nRecord = 8
country$ = "Luxembourg"
title$ = "Grand Duke"
lname$ = "Henri"

OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    PUT #1, nRecord
CLOSE #1
END
```

Viewing this file with a texteditor will reveal that Luxembourg is entered beginning with byte 491, the numerical start of record #8. To preserve space for records #5, #6, and #7, Liberty BASIC filled each byte starting from number 281 and continuing to 490 with Chr$(0). The length of the file is now 580 bytes or 8 records each containing 70 characters.

```
OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
```

```
    FileLength = LOF(#1)
CLOSE #1
    PRINT "LOF = ";FileLength
END
```

- Output

```
LOF = 560
```

Retrieving a Record

Random access files are retrieved with the GET command. As with the PUT command, the record number must be identified. Retrieving the data places each field entry of that data into the respective field variable.

```
nRecord = 2
OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    GET #1, nRecord
CLOSE #1
PRINT "Country: ";country$
PRINT "Title: ";title$
PRINT "Last Name: ";lname$
END
```

- Output

```
Country: United States of America
Title: President
Last Name: Obama
```

Using GETTRIM

The field variable will contain the full number of specified bytes, meaning all the padding spaces. This may not have been evident in the preceding examples, but will be very evident in this next example.

```
nRecord = 8

OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
```

```
    GET #1, nRecord
CLOSE #1
PRINT lname$;" is the ";title$;" of ";country$;"."
END
```

- Output

```
Henri                is the Grand Duke          of Luxembourg
            .
```

Use GETTRIM rather than GET to remove those trailing spaces.

```
nRecord = 8

OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    GETTRIM #1, nRecord
CLOSE #1
PRINT lname$;" is the ";title$;" of ";country$;"."
END
```

- Output

```
Henri is the Grand Duke of Luxembourg.
```

Editing a Record

To edit a record, simply change one or more variables and PUT the record again. Since all field variables will be written with the PUT statement, it is important that all variables contain the correct information, even if that information isn't meant to be changed.

```
country$ = "United States of America"
title$ = "President"
lname$ = "Bush"
nRecord = 2

OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    PUT #1, nRecord
CLOSE #1
```

```
OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    GET #1, nRecord
CLOSE #1
PRINT "Country: ";country$
PRINT "Title: ";title$
PRINT "Last Name: ";lname$
END
```

- Output

```
Country: United States of America
Title: President
Last Name: Bush
```

Let's put President Obama back now.

```
country$ = "United States of America"
title$ = "President"
lname$ = "Obama"
nRecord = 2

OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    PUT #1, nRecord
CLOSE #1

OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    GET #1, nRecord
CLOSE #1
PRINT "Country: ";country$
PRINT "Title: ";title$
PRINT "Last Name: ";lname$
END
```

- Output

```
Country: United States of America
Title: President
Last Name: Obama
```

Recapping

- Define the LEN of the Random Access File as the sum total number of bytes of every field.
- OPEN the Random Access File for RANDOM for both reading and writing files.
- Place the data in the unique field variables before wrting a record.
- Use the PUT command to write a record.
- Use the GET command to read the record, retrieving all bytes of the field.
- Use the GETTRIM command to read the record, with all trailing filler characters trimmed from the fields.
- Use the unique field variables to retrieve the data.
- Edit a record by rewriting that entire record.

Finding the Number of Records

The LOF() function returns the file size in bytes, which is also the same as the number of characters. Since the LEN of each record remains the same, the number of records is the LOF() divided by LEN. Remember that LEN must be followed by the number and not a variable.

```
RecordLength = 70
OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    FileLength = LOF(#1)
CLOSE #1
    nRecords = FileLength / RecordLength
    PRINT "This Random Access File contains ";nRecords;" records."
END
```

- Output

```
This Random Access File contains 8 records.
```

LOF() doesn't discriminate the number of records containing valid data from the number of records containing filler data. It is up to the programmer to determine that information. The HeadsOfState.dat record should have 8 records, but 3 of those records contain nothing more than Chr$(0) space fillers.

Finding the Number of Valid Records with GETTRIM

Use GETTRIM and a counter to determine if a record has valid entries. GETTRIM will remove both padding spaces and any filler Chr$(0)'s. If the entry only consists of spaces and or Chr$(0)'s, then the field variable results in a null, or empty, string. Valid Records are those records with at least one field that doesn't trim to null. Use a counter to count the number of valid records.

```
RecordLength = 70
```

```
nValidRecords = 0
OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    FileLength = LOF(#1)
    nRecords = FileLength / RecordLength
    FOR i = 1 to nRecords
        GETTRIM #1, i
        record$ = country$ + title$ + lname$
        PRINT: PRINT "Record No: ";i
        IF record$ <> "" THEN
            nValidRecords = nValidRecords + 1
            PRINT "Country: ";country$
            PRINT "Title: ";title$
            PRINT "Last name: ";lname$
        ELSE
            PRINT "No valid data discovered."
        END IF
    NEXT i
CLOSE #1
PRINT: PRINT "Found ";nValidRecords;" valid records out of ";
PRINT nRecords;" total records."
END
```

- Output

```
Record No: 1
Country: Japan
Title: Emperor
Last name: Akihito

Record No: 2
Country: United States of America
Title: President
Last name: Obama

Record No: 3
Country: Tajikistan
Title: President
Last name: Rahmon

Record No: 4
Country: Saudi Arabia
Title: King
Last name: Abdallah

Record No: 5
```

```
No valid data discovered.

Record No: 6
No valid data discovered.

Record No: 7
No valid data discovered.

Record No: 8
Country: Luxembourg
Title: Grand Duke
Last name: Henri

Found 5 valid records out of 8 total records.
```

Deleting a Record

There is no command to delete a random access file record. The only way to erase information is to replace the fields with null or empty strings. To erase the data in record #4, set all field variables to null.

```
    country$ = ""
    title$ = ""
    lname$ = ""
OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    PUT #1, 4
CLOSE #1

    RecordLength = 70
OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    FileLength = LOF(#1)
    nRecords = FileLength / RecordLength
    FOR i = 1 to nRecords
        GETTRIM #1, i
        record$ = country$ + title$ + lname$
        IF record$ <> "" THEN
            nValidRecords = nValidRecords + 1
        END IF
        PRINT: PRINT "Record No: ";i
        PRINT "Country: ";country$
        PRINT "Title: ";title$
```

```
        PRINT "Last name: ";lname$
    NEXT i
CLOSE #1
PRINT: PRINT "Found ";nValidRecords;" valid records out of ";
PRINT nRecords;" total records."
END
```

- Output

```
Record No: 1
Country: Japan
Title: Emperor
Last name: Akihito

Record No: 2
Country: United States of America
Title: President
Last name: Obama

Record No: 3
Country: Tajikistan
Title: President
Last name: Rahmon

Record No: 4
Country:
Title:
Last name:

Record No: 5
Country:
Title:
Last name:

Record No: 6
Country:
Title:
Last name:

Record No: 7
Country:
Title:
Last name:

Record No: 8
Country: Luxembourg
```

```
Title: Grand Duke
Last name: Henri

Found 4 valid records out of 8 total records.
```

The information can be set to null, but the record will remain. Housing many empty files will needlessly inflate the size of the file. There are several ways to recover the empty space in a random access file. One way is to simply rewrite the file to a temporary file, PUTting only those records with valid data, delete the original file with the KILL command, and then rename the temporary file to the original file name using the NAME command.

```
RAF1$ = DefaultDir$;"/HeadsOfState.dat"
RAF2$ = DefaultDir$;"/temp.dat"
RecordLength = 70
ct = 0

OPEN RAF1$ for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    FileLength = LOF(#1)
    nRecords = FileLength / RecordLength
    PRINT "Original File"
    PRINT "File Length = ";FileLength
    PRINT "Number of Records = ";nRecords
    PRINT
    OPEN RAF2$ for RANDOM as #2 LEN = 70
        FIELD #2, 30 as country$, 20 as title$, 20 as lname$
        FOR i = 1 to nRecords
            GET #1, i
            record$ = TRIM$(country$) + TRIM$(title$) + TRIM$(lname$)
            IF record$ <> "" THEN
                ct = ct + 1
                PUT #2, ct
            END IF
        NEXT i
    CLOSE #2
CLOSE #1

KILL RAF1$
NAME RAF2$ AS RAF1$

OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    FileLength = LOF(#1)
    nRecords = FileLength / RecordLength
    PRINT "New File"
```

```
    PRINT "File Length = ";FileLength
    PRINT "Number of Records = ";nRecords
    PRINT
    FOR i = 1 to nRecords
        GET #1, i
        PRINT "Record No: ";i
        PRINT "Country: ";country$
        PRINT "Title: ";title$
        PRINT "Last Name: ";lname$
        PRINT
    NEXT i
CLOSE #1
END
```

- Output

```
Original File
File Length = 560
Number of Records = 8

New File
File Length = 280
Number of Records = 4

Record No: 1
Country: Japan
Title: Emperor
Last Name: Akihito

Record No: 2
Country: United States of America
Title: President
Last Name: Obama

Record No: 3
Country: Tajikistan
Title: President
Last Name: Rahmon

Record No: 4
Country: Luxembourg
Title: Grand Duke
Last Name: Henri
```

Another way to preserve space is to look for a null record, then use that record for the next entry. If there are 6 total records, and 2 records contain no data, then PUT the new data in one of the null records rather

than adding a 9th record. Add record #6 to HeadsOfState.dat.

```
    country$ = "Aruba"
    title$ = "Governor"
    lname$ = "Refenjol"
OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    PUT #1, 6
CLOSE #1
END
```

Read the file.

```
OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    FOR i = 1 to 6
        GETTRIM #1, i
        PRINT "Record No: ";i
        PRINT "Country: ";country$
        PRINT "Title: ";title$
        PRINT "Last name: ";lname$
        PRINT
    NEXT i
CLOSE #1
END
```

- Output

```
Record No: 1
Country: Japan
Title: Emperor
Last name: Akihito

Record No: 2
Country: United States of America
Title: President
Last name: Obama

Record No: 3
Country: Tajikistan
Title: President
Last name: Rahmon

Record No: 4
```

```
Country: Luxembourg
Title: Grand Duke
Last name: Henri

Record No: 5
Country:
Title:
Last name:

Record No: 6
Country: Aruba
Title: Governor
Last name: Refenjol
```

Add another file, but this time, search for an empty record first. Then replace the empty contents with the new data. Record #5 should be empty.

```
    IsEmpty = 0
OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
    nRecords = LOF(#1) / 70
    FOR i = 1 to nRecords
        GETTRIM #1, i
        record$ = country$ + title$ + lname$
        IF record$ = "" THEN
            IsEmpty = 1
            selRecord = i
            EXIT FOR
        END IF
    NEXT i
    IF IsEmpty = 0 THEN
        selRecord = nRecords + 1
    END IF
    country$ = "Romania"
    title$ = "President"
    lname$ = "Basescu"
    PUT #1, selRecord
CLOSE #1
END
```

Rather than adding a new record, the data was placed in the previously empty record.

```
OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
    FIELD #1, 30 as country$, 20 as title$, 20 as lname$
```

```
    FOR i = 1 to LOF(#1) / 70
        GETTRIM #1, i
        PRINT "Record No: ";i
        PRINT "Country: ";country$
        PRINT "Title: ";title$
        PRINT "Last name: ";lname$
        PRINT
    NEXT i
CLOSE #1
END
```

- Output

```
Record No: 1
Country: Japan
Title: Emperor
Last name: Akihito

Record No: 2
Country: United States of America
Title: President
Last name: Obama

Record No: 3
Country: Tajikistan
Title: President
Last name: Rahmon

Record No: 4
Country: Luxembourg
Title: Grand Duke
Last name: Henri

Record No: 5
Country: Romania
Title: President
Last name: Basescu

Record No: 6
Country: Aruba
Title: Governor
Last name: Refenjol
```

Recapping

- Divide the LOF of the random access file by the number of bytes in each record to find the total number of records in the file
- Use GETTRIM to remove trailing spaces and Chr$(0) fillers
- Erase the contents of a record by setting the unique field variable strings to null and then PUTting that record.
- Conserve file size by searching for and replacing records containing null data with new data before adding a new record

Common Errors with Random Access File Programming

Errors are most often associated with problems in reading or writing a record to file.

- Runtime Error: "isEmpty" not understood
- (see error.log for more information)

will occur if the program attempts to read a non-existant record, such as if the file holds 8 records and the program attempts to retrieve information from record #12.

- Runtime Error: index: -4025 is outside of collection bounds
- (see error.log for more information)

occurs when an attempt is made to read the non-existant record #0.

- Runtime Error: copy beyond end of file
- (see error.log for more information)

will occur when attempts are made to retrieve information from a non-existant file. A program typo in the file's name may be responsible.

Probably the most serious error occurs when attempting to write to the 0 record. The "isEmpty" not understood error is the first to occur. The file itself becomes corrupted and maybe even the path to the file. You may get a file size of 4 gb or greater. The only solution is to delete the corrupt file and start over. However, attempts to delete the file often result in a Windows' "unable to access file" error and deletion becomes impossible. Other files in the same subfolder may become unaccessible as well. Sometimes closing the subfolder with Task Manager will allow the file to finally be deleted. The results of this error are unpredictable, but always fatal. The writing to record 0 error is most often produced when a variable is used to indicate the record number and the value of the variable is 0. If you ever encounter problems accessing a file, suspect this writing to record #0 as the cause. You may want to include a way to check the value of the record number variable to prevent this error.

```
    GOSUB [WriteRAFData]
END

[WriteRAFData]
    IF nRecord = 0 THEN
        NOTICE "Unable to write data to record #";nRecord
        RETURN
    END IF
    OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
        FIELD #1, 30 as country$, 20 as title$, 20 as lname$
        PUT #1, nRecord
    CLOSE #1
RETURN
```

Other errors may occur because of data being inadvertently assigned to the unique field variable names prior to the PUT statement, or variables being prematurely replaced with the GET statement. To reduce errors with variables retrieved from and written to a random access file, restrict the use of the unique field variable names to the actual file reading and writing routines.

```
    OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
        FIELD #1, 30 as country$, 20 as title$, 20 as lname$
        GETTRIM #1, 2
    CLOSE #1
    selCountry$ = country$
    selTitle$ = title$
    selLname$ = lname$
    PRINT "Country: ";selCountry$
    PRINT "Title: ";selTitle$
    PRINT "Last name: ";selLname$
END
```

- Output

```
Country: United States of America
Title: President
Last name: Obama
```

Be sure to use the variables selCountry$, selTitle$, and selLname$ throughout the main portion of the program. When it's time to write the record, transfer the data to the unique field variable names. The following code contains error prevention for both PUTting and GETting records of a random access file.

```
    selCountry$ = "Spain"
    selTitle$ = "King"
```

```
    selLname$ = "Carlos"
    nRecord = 7
    country$ = selCountry$
    title$ = selTitle$
    lname$ = selName$
    GOSUB [WriteRAFData]
    FOR nRecord = 1 to 7
        GOSUB [ReadRAFData]
        PRINT "Record #";nRecord
        PRINT "Country: ";selCountry$
        PRINT "Title: ";selTitle$
        PRINT "Last name: ";selLname$
        PRINT
    NEXT nRecord
END

[WriteRAFData]
    IF nRecord = 0 THEN
        NOTICE "Cannot write to Record #";nRecord
        RETURN
    END IF
    OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
        FIELD #1, 30 as country$, 20 as title$, 20 as lname$
        PUT #1, nRecord
    CLOSE #1
RETURN

[ReadRAFData]
    IF nRecord = 0 THEN
        NOTICE "Cannot retrieve from Record #";nRecord
        RETURN
    END IF
    OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
        FIELD #1, 30 as country$, 20 as title$, 20 as lname$
        NumberRecords = LOF(#1) / 70
    CLOSE #1
    IF nRecord > NumberRecords THEN
        NOTICE nRecord;" doesn't exist."
        RETURN
    END IF
    OPEN "HeadsOfState.dat" for RANDOM as #1 LEN = 70
        FIELD #1, 30 as country$, 20 as title$, 20 as lname$
        GETTRIM #1, nRecord
    CLOSE #1
    selCountry$ = country$
    selTitle$ = title$
```

```
    selLname$ = lname$
RETURN
```

Advantages and Disadvantages of a Random Access File

Random access files are useful for storing records when those records contain similar information. Random access files provide quick access to information and is efficient for retrieving information from only one or a few records at a time. Unlike sequential access files, only the modified information needs to be rewritten. Random access files are a bit more complex than sequential text files. Care must be taken that the program maintains the uniqueness of the field variable names or unintended changes could be made.

More on Files

An Introduction to Working with Files
Random Access Files (Jim Brossman) LB Newsletter #120