Encrypt and decrypt functions

Here is some code that you can use in a `.pas` file that was written using encryption. Save this information to a unit called `UEncrypt.pas`.

The key here is creating a Seed Key.

```
unit UEncrypt;

interface

function Decrypt(const S: AnsiString; Key: Word): AnsiString;
function Encrypt(const S: AnsiString; Key: Word): AnsiString;

implementation

const
  C1 = 52845;
  C2 = 22719;

function Decode(const S: AnsiString): AnsiString;
const
  Map: array[Char] of Byte = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 62, 0, 0, 0, 63, 52, 53,
    54, 55, 56, 57, 58, 59, 60, 61, 0, 0, 0, 0, 0, 0, 0, 0, 1, 2,
    3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
    20, 21, 22, 23, 24, 25, 0, 0, 0, 0, 0, 0, 26, 27, 28, 29, 30,
    31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45,
    46, 47, 48, 49, 50, 51, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0
  );
var
  I: LongInt;
begin
  case Length(S) of
    2:
    begin
      I := Map[S[1]] + (Map[S[2]] shl 6);
      SetLength(Result, 1);
      Move(I, Result[1], Length(Result))
    end;
    3:
    begin
      I := Map[S[1]] + (Map[S[2]] shl 6) + (Map[S[3]] shl 12);
      SetLength(Result, 2);
      Move(I, Result[1], Length(Result))
    end;
    4:
    begin
      I := Map[S[1]] + (Map[S[2]] shl 6) + (Map[S[3]] shl 12) +
        (Map[S[4]] shl 18);
      SetLength(Result, 3);
      Move(I, Result[1], Length(Result))
    end;
  end;
end;

function PreProcess(const S: AnsiString): AnsiString;
var
  SS: AnsiString;
begin
  SS := S;
  Result := '';
  while SS <> '' do
```

```
  begin
    Result := Result + Decode(Copy(SS, 1, 4));
    Delete(SS, 1, 4)
  end;
end;

function InternalDecrypt(const S: AnsiString; Key: Word): AnsiString;
var
  I: Word;
  Seed: Word;
begin
  Result := S;
  Seed := Key;
  for I := 1 to Length(Result) do
  begin
    Result[I] := Char(Byte(Result[I]) xor (Seed shr 8));
    Seed := (Byte(S[I]) + Seed) * Word(C1) + Word(C2)
  end;
end;

function Decrypt(const S: AnsiString; Key: Word): AnsiString;
begin
  Result := InternalDecrypt(PreProcess(S), Key)
end;

function Encode(const S: AnsiString): AnsiString;
const
  Map: array[0..63] of Char = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ' +
    'abcdefghijklmnopqrstuvwxyz0123456789+/';
var
  I: LongInt;
begin
  I := 0;
  Move(S[1], I, Length(S));
  case Length(S) of
    1:
      Result := Map[I mod 64] + Map[(I shr 6) mod 64];
    2:
      Result := Map[I mod 64] + Map[(I shr 6) mod 64] +
        Map[(I shr 12) mod 64];
    3:
      Result := Map[I mod 64] + Map[(I shr 6) mod 64] +
        Map[(I shr 12) mod 64] + Map[(I shr 18) mod 64]
  end;
end;

function PostProcess(const S: AnsiString): AnsiString;
var
  SS: AnsiString;
begin
  SS := S;
  Result := '';
  while SS <> '' do
  begin
    Result := Result + Encode(Copy(SS, 1, 3));
    Delete(SS, 1, 3)
  end;
end;

function InternalEncrypt(const S: AnsiString; Key: Word): AnsiString;
var
  I: Word;
  Seed: Word;
begin
  Result := S;
  Seed := Key;
  for I := 1 to Length(Result) do
  begin
    Result[I] := Char(Byte(Result[I]) xor (Seed shr 8));
    Seed := (Byte(Result[I]) + Seed) * Word(C1) + Word(C2)
  end;
end;

function Encrypt(const S: AnsiString; Key: Word): AnsiString;
```

```
begin
  Result := PostProcess(InternalEncrypt(S, Key))
end;

end.
```

To demonstrate, create a windows form with three *TMemo* controls and a *TButton* and add a button click event handler as follows:

```
uses
  ...
  UEncrypt;
...
procedure TForm1.Button1Click(Sender: TObject);
const
  SeedKey = 53269;
begin
  Memo2.Text := Encrypt(Memo1.Text,SeedKey);
  Memo3.Text := Decrypt(Memo2.Text,SeedKey);
end;
```

Enter some text to be converted into *Memo1*. Click the button. The encrypted code appears in *Memo2* and the decrypted code in *Memo3*.

The demo code and narrative have been tweaked a little.

Author:	*Steve Schafer*
Added:	2009-06-23
Last updated:	2009-08-11