# Encrypting a text string based on a key in Delphi

There are lots of way can encrypt a text string based on a key. By the approach, you can use a single numerical key, or text string. In terms of security's point of view, text string-based key has better security performance.Below code is an example of string encryption module.

```
function Encrypt(const InString:string; Salt:string): string;
var
  i : Byte;
  StartKey, MultKey, AddKey: Word;
begin
  Result := '';
  if (Salt = '') then begin
    Result := InString;
  end
  else begin
    StartKey := Length(Salt);
    MultKey := Ord(Salt[1]);
    AddKey := 0;
    for i := 1 to Length(Salt) - 1 do AddKey := AddKey + Ord(Salt[i]);
    for i := 1 to Length(InString) do
    begin
      Result := Result + CHAR(Byte(InString[i]) xor (StartKey shr 8));
      StartKey := (Byte(Result[i]) + StartKey) * MultKey + AddKey;
    end;
  end;
end;

function Decrypt(const InString:string; Salt:string): string;
var
  i : Byte;
  StartKey, MultKey, AddKey: Word;
begin
  Result := '';
  if (Salt = '') then begin
    Result := InString;
  end
  else begin
    StartKey := Ord(Salt[1]);
    MultKey := Length(Salt);
    AddKey := 0;
    for i := 1 to Length(Salt) - 1 do AddKey := AddKey + Ord(Salt[i]);
    for i := 1 to Length(InString) do
    begin
      Result := Result + CHAR(Byte(InString[i]) xor (StartKey shr 8));
      StartKey := (Byte(InString[i]) + StartKey) * MultKey + AddKey;
    end;
  end;
end;
```

By the way, above was not helpful for me - I use multibyte character, and below code was so helpful even though I can't use any key on the documentation

```
uses IdCoder, IdCoderMIME, IdGlobal;

..
..

begin
    // encode string
    sMasterXML.Text := TIdEncoderMIME.EncodeString(sMasterXML.Text, IndyTextEncoding_UTF8);

        // deocde string
    sMasterXML.Text := TIdDecoderMIME.DecodeString(sMasterXML.Text, IndyTextEncoding_UTF8);

end;
```

Below is combined encryption algorithm which provides much better secure results.

```
uses IdCoder, IdCoderMIME, IdGlobal;

.
.
.


const CK_ENC_VAR_RANGE=10;

function CKEncrypt(const Source:string; Salt:string): string;
var
  i, lenSalt, key, keyWeight: Word;
  InString: string;
begin
  Result := '';
  if (Salt = '') then begin
    Result := Source;
  end else begin
    InString := TIdEncoderMIME.EncodeString(Source, IndyTextEncoding_UTF8);

    lenSalt := Length(Salt);

    keyWeight := 0;
    for i := 1 to lenSalt do keyWeight := keyWeight + ord(Salt[i]);
    keyWeight := keyWeight mod CK_ENC_VAR_RANGE;

    for i := 1 to Length(InString) do
    begin
      key := ( ord(Salt[(i mod lenSalt) + 1]) mod CK_ENC_VAR_RANGE);
      Result := Result + CHAR(Byte(InString[i]) + key - keyWeight);
    end;
  end;
end;

function CKDecrypt(const InString:string; Salt:string): string;
var
  i, lenSalt, key, keyWeight: Word;
  Target: string;
begin
  if (Salt = '') then begin
    Target := InString;
  end else begin
    lenSalt := Length(Salt);

    keyWeight := 0;
    for i := 1 to lenSalt do keyWeight := keyWeight + ord(Salt[i]);
    keyWeight := keyWeight mod CK_ENC_VAR_RANGE;

    Target := '';
    for i := 1 to Length(InString) do
    begin
      key := ( ord(Salt[(i mod lenSalt) + 1]) mod CK_ENC_VAR_RANGE);
      Target := Target + CHAR(Byte(InString[i]) - key + keyWeight);
    end;
    Result := TIdDecoderMIME.DecodeString(Target, IndyTextEncoding_UTF8);
  end;
end;

{ Testng }
procedure TForm1.Button1Click(Sender: TObject);
begin
  Memo2.Text := CKEncrypt(Memo1.Text, 'foo');
  Memo3.Text := CKDecrypt(Memo2.Text, 'foo');
end;
```