Lær at lave dine egne programmer

Det er lettere at lave sine egne programmer, end man skulle tro. Med et moderne programmeringsværktøj som Delphi 6.0 kan du lave rigtige programmer i løbet af kort tid.

Har du nogen sinde overvejet at lave dine helt egne programmer? Det er faktisk ikke så svært, som det lyder. I gamle dage var programmering et frygtelig avanceret arbejde, hvor man sad og skrev side op og side ned med kryptiske koder. Koderne er der stadig. Men det meste arbejde sker i dag ved at trække og slippe komponenter med musen. Det meste af koden bliver lavet automatisk, og det besværlige arbejde med at lave flotte brugerflader er blevet gjort enkelt – det er ikke sværere end at bruge et tegneprogram.

I denne artikel viser vi dig, hvordan du laver et lille spil. Du behøver ikke nogen forkundskaber for at prøve selv, og vi har lagt et programmeringsværktøj med på K-CD'en, som du kan bruge til formålet.

Moderne visuelle programmeringssprog, som det vi benytter i denne artikel, fungerer ved hjælp af såkaldte komponenter. De kan fx hedde knapper, indtastningsfelter og listbokse. Til hver komponent kan man knytte små stumper programkode, som afvikles, når man påvirker dem – fx ved at klikke på en knap. Selv om det lyder kompliceret, er denne teknik faktisk med til at gøre programmerne langt mere overskuelige.

Selve koden er dog grundlæggende den samme, som den har været i mange år. I denne artikel kan du lære de grundlæggende begreber i al programmering. Det gør dig ikke til ekspert på én gang, men med lidt træning kan du hurtigt lære at lave dine egne programmer. ■

Variablernes verden

Variabler bruges i næsten alle programmer som en slags »hukommelse«, hvor man kan opbevare værdier.

Hvis man skal indtaste et tal eller noget tekst i et program, vil det blive opbevaret i en variabel. Og når det lille gættespil, som du lærer at lave i denne artikel, laver et tilfældigt tal, som du kan gætte, bliver tallet også anbragt i en variabel.

Variablen genkendes på dens navn, som skrives i programmet. Før man kan bruge en variabel i sit program, skal man *erklære* den. Det vil sige, at man skal fortælle computeren, at den skal reservere plads i hukommelsen til variablen. Samtidig skal programmet vide, hvilken type variabel der er tale om. Altså om det er tekst, heltal, brøktal eller noget fjerde. I vores program bruger vi mest heltal, som i programmør-jargon kaldes for »integer«. I Delphi erklærer vi derfor vores variabel således:

```
alder :Integer;
```

Denne linie fortæller computeren, at den skal oprette en variabel med navnet »alder« – og af typen heltal. Nu kan vi bruge variablen i vores program. For at give variablen en værdi skriver vi:

```
alder := 20;
```

Herefter forbindes variablen »alder« med værdien 20. Hvis vi nu bliver et år ældre, skal alderen forøges med en. Dette gøres således:

```
alder := alder + 1;
```

Nu er værdien af alder 21. Metoden er at sætte variablen lig med dens eget indhold – plus en. Ovenstående er alt, hvad du får brug for i vores lille program. Og det rækker et godt stykke, når du skal lave egne programmer.

Programmers indre logik

Traditionelt består programmer af en masse linier programkode, som er skrevet i et bestemt programmeringssprog. Linierne afvikles én ad gangen og bestemmer, hvordan programmet fungerer. Her er nogle af de vigtigste:

Betingelser
Betingelser er noget af det vigtigste i ethvert programmeringssprog. Betingelser gør det muligt at tjekke værdierne i programmets variabler og ud fra resultatet bestemme, hvad programmet skal gøre derefter. Forestil dig et program, som skal undersøge alderen for en person, som skal ind og se en gyserfilm i biografen. Alderen på personen er gemt i variablen »alder«, og grænsen er 18 år, så programmet må derfor kun lukke personen ind i biografen, hvis vedkommende er over 18:

```
if alder >= 18 then
    // luk personen ind i biografen
else
    // giv personen en afvisning
```

Det vigtige er første og tredje linie. I et rigtigt program skal de andre linier erstattes af rigtig kode – for eksempel hvis programmet skal udskrive afvisningen på skærmen.

Løkker
Der findes flere slags løkker, men fælles er, at de afvikler en del af koden flere gange, indtil en bestemt opfyldelse er nået. Man kan fx lave et program, som skal sætte et lys i en digital lagkage for hvert år, som en person har levet. Det kan laves med en såkaldt »for«-løkke:

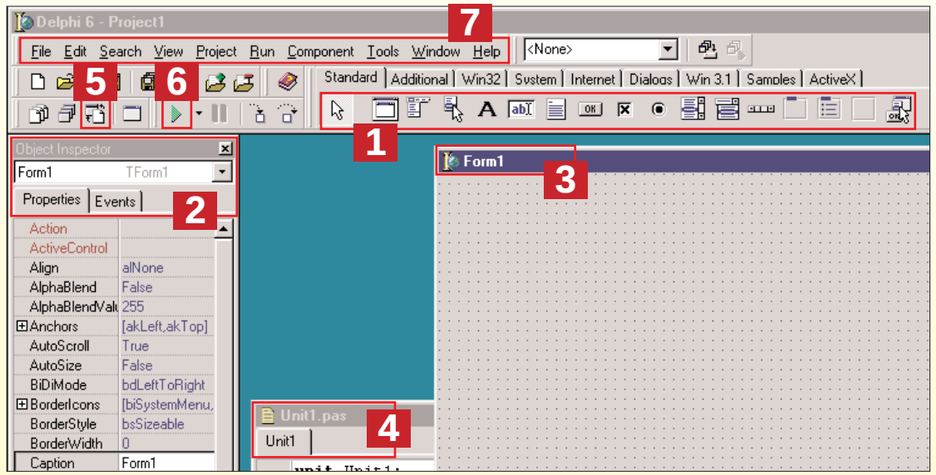
```
for taeller := 0 to alder do
    // Sæt et lys i lagkagen
```

Denne metode kræver en ekstra variabel – taeller – for at fungere. Der tælles i dette eksempel fra 0 og op til værdien gemt i variablen »alder«.

Tegn dine programmer direkte på skærmen

Sådan fungerer de forskellige dele af Delphi 6.0.

På K-CD'en finder du et af de simpleste og mest populære programmerings-værktøjer, der findes, Delphi 6.0. Der er vel at mærke tale om en fuld version af programmet. Uden begrænsninger af nogen slags.



Delphi er et visuelt programmeringsværktøj – det vil sige, at det meste af »programmeringen« sker ved at tegne figurer på skærmen og klikke med musen. Det gør sproget meget lettere at gå til – også for begyndere.

1 Komponentmenu
De mange forskellige komponenter, som du kan bruge i Delphi, findes på komponent-menuen. Hvert faneblad dækker over en række komponenter, men i vores tilfælde vil vi kun koncentrere os om dem, du kan finde under »Standard«. Når du har valgt en komponent med musen, placeres den på brugerfladen ved at klikke på brugerfladen med venstre museknap.

2 Object Inspector
Egenskaberne for de enkelte komponenter i dit program indstilles i »Object Inspector«-vinduet. Når du har valgt en komponent på din brugerflade med musen, kan du se dens egenskaber i »Object Inspector«. Menuen har to faneblade – »Properties«, hvor du foretager ændringer i egenskaber, og »Events«, hvor man bestemmer de funktioner, som skal aktiveres, når brugeren gør noget.

3 Form
Programmets brugerflade, altså det billede, som brugeren får på skærmen, tegnes i det såkaldte »Form«-vindue. Komponenter placeres med musen, hvor du ønsker dem. Selve programmeringen tilføjes senere. Når programmet startes, er det indholdet af »Form«-vinduet, som brugeren vil få at se.

4 Kodevinduet
Selve programmet findes i kodevinduet. Når du placerer komponenter på brugerfladen, bliver ændringer automatisk tilføjet i kodevinduet. Det er også her, du selv indtaster de dele af programmet, som ikke bliver lavet automatisk. Fx den del, som tjekker, om gættet er for højt eller for lavt.

5 Skift-knappen
Med denne knap skifter du mellem kodevinduet og »Form-vinduet«. Derfor behøver du ikke at kunne se vinduet for at få fat i det. Bare skift mellem de to vinduer ved at klikke på denne knap. Du kan også skifte mellem vinduerne ved at trykke på F12-tasten.

6 Play-knappen
Når du vil starte programmet, sker det ved hjælp af »Play«-knappen. Når programmet kører, kan du stoppe det ved at klikke på »Luk«-ikonet i øverste højre hjørne af vinduet. Herefter vil du blive sendt tilbage i udviklingsmiljøet. Du kan altid se, om et program kører, ved at se på »Play«-knappen.

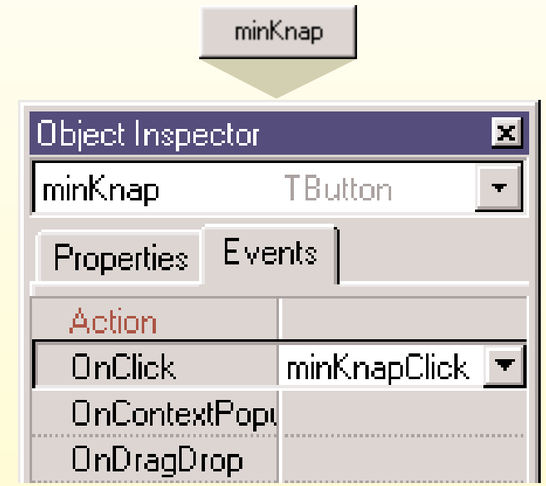
7 Menuerne
De lidt mere eksotiske funktioner i Delphi er gemt i menuerne. Herfra kan du gemme og hente projekter, du har lavet, osv. Værd at nævne er også muligheden for at nulstille et program under afvikling, hvis der sker en fejl. Bare klik på »Run«-menuen, og vælg »Program Reset« for at stoppe afviklingen af programmet og returnere til Delphi.

Delphi er sammensat af komponenter

Programmeringen foregår ved hjælp af små moduler.

Moderne, visuelle programmeringsværktøjer som Delphi er primært baseret på komponenter. Det gør programmørens arbejde lettere, når et program opbygges af mange små enkeltdele i stedet for at være en endeløs stribe af programkoder.

En komponent kan fx være en knap, et indtastningsfelt eller lignende. En komponent har en række egenskaber knyttet til sig, som i virkeligheden bare er variabler, der fortæller noget om, hvordan komponenten skal se ud og opføre sig.



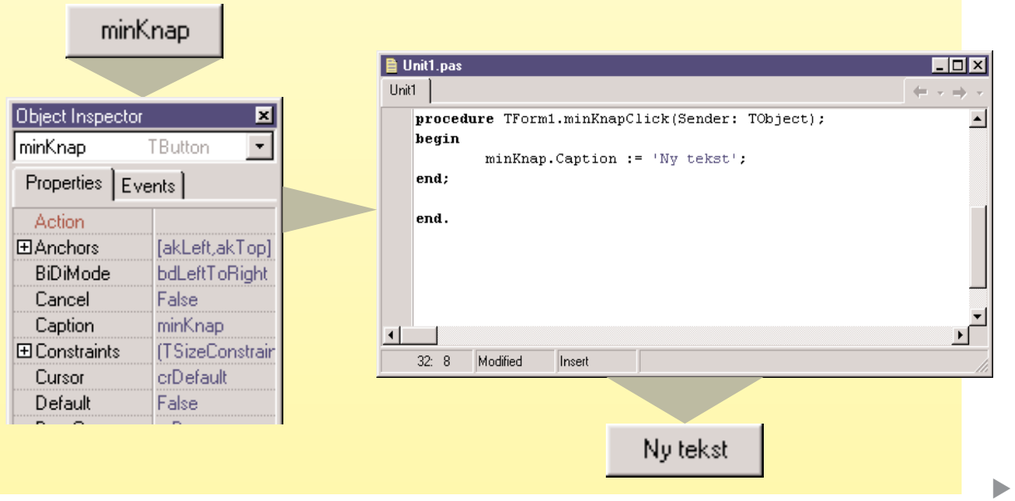
Foruden at indeholde parametre og variabler kan komponenter sende beskeder til hinanden. Det sker ved hjælp af de såkaldte »events«. En event er en besked, som sendes af komponenten, når brugeren foretager en handling. Hvis der klikkes på en knap, udsender knappen en event, som fortæller, at knappen er blevet klikket. Eventen kan bruges til at starte andre funktioner.

Et eksempel: Hvis vi dobbeltklikker på en knap på vores brugerflade, oprettes der automatisk en funktion, som køres, når man klikker på knappen. Funktionen består af en programkode, som bestemmer, hvad der skal ske, når man klikker på knappen.

Hver komponent har et navn, som man kan henvise til inde fra selve programmet. Ved at ændre indstillinger for komponenter kan man faktisk ændre programmets udseende, mens det kører. Vi kaldte vores knap for »minKnap«. Hvis vi vil ændre den tekst, som står på knappen, skriver vi følgende:

```
minKnap.Caption := 'Ny tekst';
```

Først skriver vi navnet på komponenten (minKnap), derefter et punktum og navnet på den variabel eller egenskab, som vi vil ændre i komponenten. I dette tilfælde er det »Caption«-feltet, som indeholder den tekst, som skal stå på selve knappen. Knappen får påskriften »Ny tekst«. Når programmet afvikles, og brugeren trykker på knappen, ændres teksten på knappen, og den nye tekst vises.



Sluk pc'en, og skriv programmet på papir

Den første del af programmeringen sker med blyant.

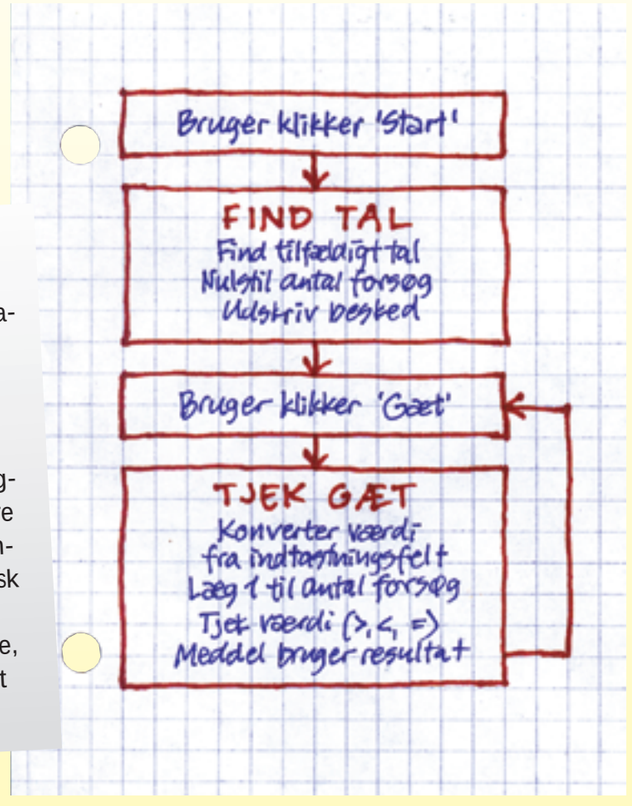
Nu er det tid at lave vores første lille program. Vi har valgt at lave et spil over en gætteleg, som du nok kender – »Gæt et tal«. Det er lettere, end man skulle tro. Bare følg vejledningen på de næste sider, så har du snart lavet dit første rigtige program.

1 Bestem først, hvad programmet skal. Inden vi går i gang med at programmere, er det en god idé at lave en lille plan på papir først. Den skal indeholde information om, hvordan programmet skal virke, og hvordan vi har forestillet os, at brugerfladen skal se ud. Vi starter med at beskrive vores lille spil:

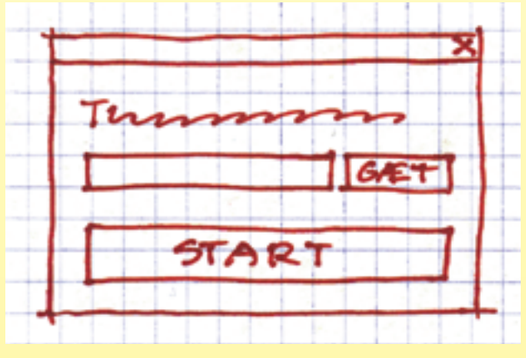
Gæt et tal

Spillet »Gæt et tal« skal fungere, ved at programmet automatisk laver et tilfældigt tal, når spilleren klikker på en knap. Herefter kan han indtaste et tal i et indtastningsfelt og med en knap afprøve, om tallet i feltet er det rigtige. Er gættet større eller mindre end det rigtige tal, skal programmet fortælle det. Er tallet identisk med det tilfældige tal, skal programmet bekræfte det og skrive, hvor mange forsøg der er brugt for at finde det rigtige tal.

2 Foruden beskrivelsen er det en god idé at lave et diagram over, hvordan programmet skal virke. Vores ser således ud:



3 Der er fire elementer på vores brugerflade: Et tekstfelt, hvor programmet skriver meddelelser til brugeren, et indtastningsfelt, hvor brugeren kan skrive et tal, en knap, hvor han kan afprøve, om han har gættet rigtigt, og en start-knap til at generere et nyt tilfældigt tal. Brugerfladen skal se nogenlunde således ud:



4 Nu er vi færdig med planlægningen af programmet. Så er det bare om at komme hen til computeren, starte Delphi og skrive dit program.

Kan du tegne, kan du også programmere

Programmets udseende tegnes som i et tegneprogram.

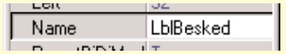
Når du starter Delphi, laver programmet et nyt projekt helt automatisk. Projektet er den »skabelon«, som et program bygges op over. Vi lægger ud med at lave vores brugerflade, altså den del af programmet, som man får at se på skærmen.

1 Først skal vi have anbragt en »Label«-komponent, som skal bruges til at sende meddelelser. Vælg »Label« fra komponentmenuen, og anbring den øverst på vores »Form«. Nu skal de enkelte parametre tilpasses. Start med at klikke på din label.

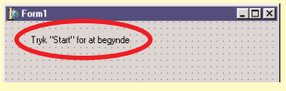
I »Object Inspector« sætter du følgende parametre:

Caption: Tryk på »Start« for at begynde

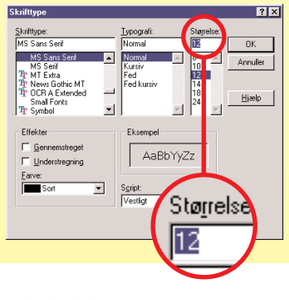
Name: LblBesked



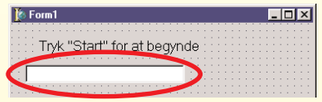
»Caption« er den tekst, som vises på skærmen, og »Name« er navnet på vores komponent, som bruges inde fra selve programmet. Bogstaverne »Lbl« er med i navnet, så man lettere kan se, at denne komponent er af typen »Label«.



Til sidst vil vi gøre skrifttypen lidt større. Klik på ordet »Font« i »Object Inspector«. Bemærk, at der kommer en lille knap til syne med tre prikker i det hvide felt til højre. Klik på den for at åbne font-menuen. Sæt størrelsen til »12«, og klik på »OK«.



2 Brugeren skal have mulighed for at indtaste et tal i programmet. Derfor skal vi bruge et indtastningsfelt. Fra komponentmenuen vælger du »Edit«. Anbring indtastningsfeltet under den »label«, som du placerede lige før. Nu kan man indtaste sit gæt.

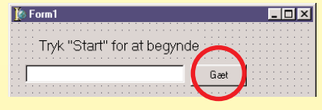


Sæt derefter følgende parametre i »Object Inspector«:

Name: EdtGaet
Text:

Vi har givet gætte-feltet navnet »EdtGaet«. »Edt« betyder »edit«. »Gaet« skyldes, at man ikke må bruge Æ, Ø og Å andre steder end i »Caption«-felter. Der skal ikke stå noget ud for »Text« fra starten. Når man senere skriver i indtastningsfeltet, bliver det opbevaret her.

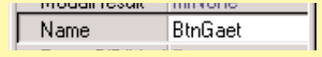
3 Til sidst skal vi have anbragt et par knapper. Den ene benyttes, når brugeren vil starte et nyt spil. Den anden, når brugeren vil gætte på et tal. Vi starter med den sidste af dem. Vælg »Button« fra komponentmenuen, og anbring den til højre for indtastningsfeltet.



Derefter sættes parametrene i »Object Inspector«:

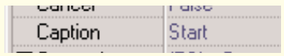
Caption: Gæt

Name: BtnGaet

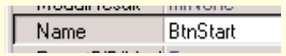


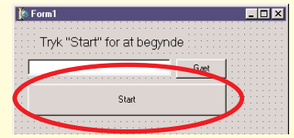
Sidste knap skal anbringes i bunden af vores »Form«. Proceduren er ligesom foregående. Her er parametrene blot:

Caption: Start



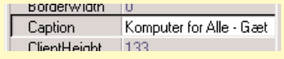
Name: BtnStart



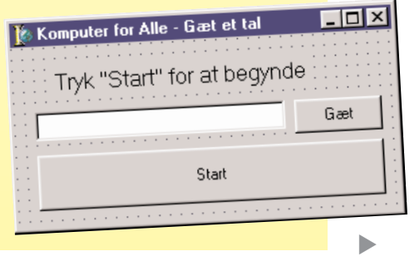


Nu er brugerfladen næsten færdig. Du kan trække i hjørnet af vinduet for at tilpasse størrelsen, så det hele passer, som det skal. Det eneste, du mangler, er at give selve vinduet en titel. Klik derfor et eller andet sted på baggrunden. I »Object Inspector« kan du derefter skrive følgende:

Caption: Komputer for Alle – Gæt et tal



Herunder kan du se, hvordan det endelige resultat helst skal se ud. Tjek lige en ekstra gang, at alle de enkelte komponenter på skærmen har de rigtige indstillinger. Det gør du ved at klikke på dem – så vises deres parametre i »Object Inspector«-vinduet.



Sådan skriver du selve gætte-programmet

Der skal stadig laves programkode – men det er let.

Nu, hvor programmets udseende er på plads, skal du lave den programkode, som får det hele til at fungere. Det meste af koden kommer faktisk af sig selv, når du klikker på komponenterne. Men der skal stadig fyldes et par linier ind. Heldigvis hjælper Delphi også her.

For at dit program skal fungere korrekt, er det vigtigt, at du skriver programkoden, præcis som den står i bladet. Du skal især holde øje med, at alle semikoloner og mellemrum kommer med.

DEFINÉR VARIABLER

1 Først skal vi have defineret vores variabler. De indeholder de data, som programmet skal behandle. I vores tilfælde drejer det sig om to værdier: Det tilfældige tal, brugeren skal gætte, og det antal forsøg, som han har brugt på at gætte tallet. Nu er det tid at skifte til det vindue, hvor vi har vores kode. Tryk på F12-tasten for at se koden.

Faktisk er en del af programmet allerede lavet. Det skete automatisk, da vi lavede vores brugerflade. Det meste af det behøver du ikke tænke på nu. Men vi skal have indsat vores variabler i programmet. For at hjælpe os har Delphi allerede markeret området med blå tekst. Vores variabler skal indsættes på linien efter teksten »Private Declarations«. Da begge variabler er heltal – også kaldet »Integer« – skal du blot skrive følgende to linier, hvor der står »Private Declarations«:

```
  procedure BtnGaetClick(Sender:
private
  { Private declarations }
  tilfaeldig: Integer;
  forsoeg: Integer;
public
  { Public declarations }
end;
```

LAV ET TILFÆLDIGT TAL, OG NULSTIL ANTAL FORSØG

2 Nu har vi erklæret vores variabler i programmet, og det er tid at koncentrere sig om selve funktionaliteten – altså det, der sker, når brugeren foretager en eller anden handling.

Det mest logiske er at begynde med Start-knappen, som sætter programmet i gang. Skift tilbage til brugerfladen ved at trykke på F12-tasten. Dobbeltklik med musen på Start-knappen. Du kommer nu tilbage til kodevinduet. Delphi har automatisk lavet den funktion, som udføres, når en bruger klikker på Start-knappen. Det eneste, vi behøver, er at skrive selve programlinierne imellem ordene »begin« og »end«.

Programmet skal lave et tilfældigt tal, som brugeren skal gætte. Til formålet findes der en tilfældighedsgenerator indbygget i enhver computer. Vi sætter den først i gang ved at skrive »Randomize« i første linie af programmet. Herefter skal vi have lavet en tilfældig værdi og gemme den i en variabel, som kan bruges i hele programmet. Variablerne oprettede vi lige før, så nu er det bare at tilføje den tilfældige værdi til vores variabel. Det gøres ved hjælp af funktionen »Random()«.

Skriv følgende for at generere det tilfældige tal:

```
randomize;
tilfaeldig := Random(20);
```

Tallet i parentesen efter »Random«-kommandoen indikerer, at det tilfældige tal ligger mellem 1 og 20. Hvis du syntes, at det er for let, kan du eventuelt skrive et højere tal.

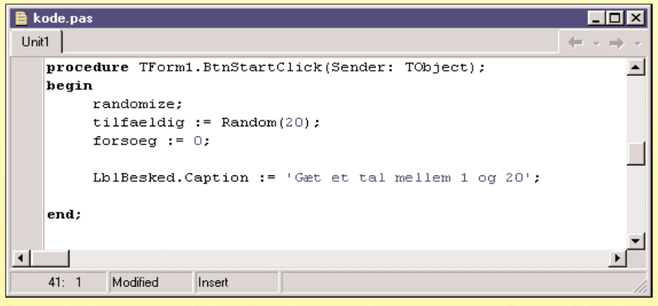
Vi skal også nulstille det antal forsøg, som brugeren foreløbig har brugt til at gætte det nye tal. Det gøres meget enkelt:

```
forsoeg := 0;
```

Endelig skal vi have skrevet en tekst til brugeren, som fortæller ham, hvad der nu skal ske. Vi bruger vores »label« til det formål. Det eneste, vi behøver, er at erstatte teksten. Som du måske kan huske, kaldte vi den »LblBesked«. For at skrive en ny tekst skal »Caption« for vores label ændres. Det gøres sådan:

```
LblBesked.Caption :=   'Gæt et tal
                       mellem 1 og 20';
```

3 Nu er vi færdige med den del af programmet. Når brugeren trykker på Start-knappen, laver programmet et tilfældigt tal, nulstiller antallet af forsøg, som er brugt på at gætte tallet, og skriver til sidst en besked, så brugeren ved, at alt er parat til et nyt spil. Hele molevitten skal gerne se således ud:



Denne kode gemmer sig bag Start-knappen i vores program. For at skrive den ind skal du først dobbeltklikke på Start og derefter udfylde det manglende.

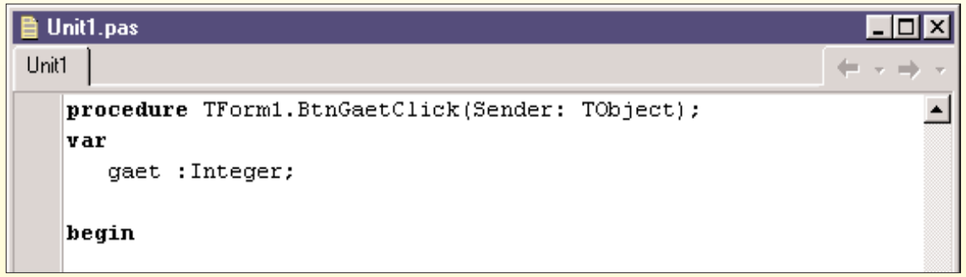
MODTAG BRUGERENS GÆT

4 Nu er vores program i stand til at starte et nyt spil og vælge et nyt tilfældigt tal. Nu skal vi lave den funktion, som undersøger brugerens indtastning og sammenligner den med det tilfældige tal.

Når brugeren har indtastet en værdi i indtastningsfeltet og klikket på »Gæt«-knappen, skal programmet undersøge, om det indtastede tal er det rigtige, eller om brugerens gæt er for højt eller lavt. I alle tilfælde skal brugeren have at vide af programmet, om gættet var rigtigt.

Som før skal vi tilbage til selve brugerfladen med F12-tasten eller knappen. Dobbeltklik nu på knappen »Gæt«. Herefter havner vi igen inde i selve kodevinduet og kan skrive den sidste del af vores program. Denne del af programmet bliver udført, når brugeren klikker på »Gæt«. Når brugeren har klikket »Gæt«, er værdien af indtastningsfeltet gemt som tekst. Men da vi skal bruge et tal, er vi nødt til at konvertere tekstværdien til et heltal. Dertil kræves endnu en variabel. Det er bedst at lave en såkaldt »lokal variabel«, som kun kan bruges i denne funktion. Som altid skal vi huske at erklære variabler, før vi bruger dem. Det gøres mellem funktionens første og anden linie, lige inden kommandoen »begin«. Ordet »var« fortæller computeren, at vi vil erklære en variabel. Herefter kommer erklæringen:

```
var
   gaet :Integer;
```



Inden programmet kan begynde at regne på, om brugeren har gættet rigtigt, skal gættet »oversættes« fra tekst til tal. Til det formål opretter vi en såkaldt lokal variabel, som kun kan bruges i denne del af programmet.

5 Nu har vi oprettet en variabel, der hedder »gaet«. Herefter skal værdien i indtastningsfeltet omsættes til tekst og gemmes i denne variabel. Dette skrives efter kommandoen »begin« således:

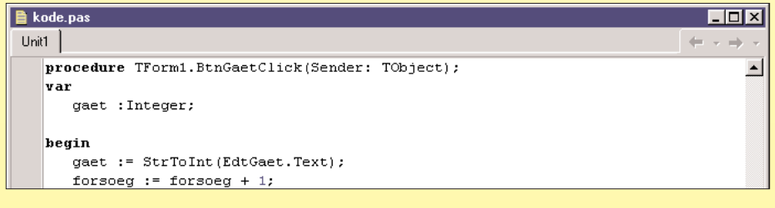
Sådan ser den kode ud, som kræves for at hente brugerens gæt, oversætte teksten til en talværdi og lægge 1 til antallet af gæt.

```
begin
  gaet := StrToInt(EdtGaet.Text);
  forsoeg := forsoeg + 1;
```

Når brugeren har indtastet sit gæt, skal det optælles. Det sker ved at tage den variabel, kaldet »forsoeg«, som tæller antallet af gæt, og forøge den med 1. Forøgelsen sker ved at lade variablen være lig med sig selv – plus en.

EdtGaet er, som nævnt, vores indtastningsfelt. Værdien af indtastningen opbevares i »Text«. Når man skriver »EdtGaet.Text«, returneres teksten i indtastningsfeltet. Uden om dette er skrevet en StrToInt()–kommando. Denne kommando ændrer teksten til et heltal. Og da vi har sat vores nye variabel »gaet« til at være lig med resultatet af StrToInt()-kommandoen, lagres heltallet her.

Det ser ganske kompliceret ud. Men den slags manøvrer er nødvendige i programmering, fordi forskellige variabeltyper ikke kan sammenlignes med hinanden uden videre. Du skal i øvrigt være opmærksom på, at der ikke er nogen fejlkontrol i vores program. Så hvis man indtaster andet end tal, opstår der fejl.

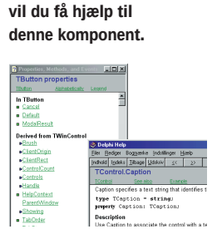


På næste side vil vi fortsætte med at sammenligne den indtastede værdi, som vi har i variablen »gaet«, med den værdien i variablen »tilfaeldig«.

Brug online-hjælp!

Selv om Delphi er relativt let, vil du uværgeligt komme ud for situationer, hvor du gerne vil vide, hvorfor dette eller hint ikke virker. Den bedste hjælp er den online-hjælp, som følger med Delphi. Den kan aktiveres på flere måder.

Hvis du klikker på en komponent på brugerfladen eller komponentmenuen og derefter trykker på F1-tasten, vil du få hjælp til denne komponent.



Du kan også vælge Hjælp-menuen og klikke på »Contents«. Når vinduet kommer frem, vælger du »Søg«. Her kan du søge efter nøgleord i hele hjælpesystemet.

TJEK GÆTTET, OG GIV ET SVAR

6 Herefter bruger vi tre »if«-kommandoer. De skal checke, om »gaet« er større end, mindre end eller lig med »tilfaeldig«. Den første if-sætning ser således ud:

```
if gaet = tilfaeldig then 1
   LblBesked.Caption := 'Du gættede tallet i ' + IntToStr(forsoeg) + '. forsøg!'; 2
```

De to andre if-sætninger fungerer på samme måde. Her skal du bare skrive brugerens gæt ud sammen med teksten i stedet for antallet af forsøg. Når du er helt færdig, skulle det gerne se således ud 3:

Lad os starte med at se på den første linie 1. Der står oversat til normalt sprog: »HVIS brugerens gæt er lig med vores tilfældige tal, SÅ«. Det, der sker, er, at hvis variablen »gaet« er lig med »tilfaeldig«, udføres den efterfølgende linie. Er de ikke ens, fortsætter programmet blot.

Den efterfølgende linie 2 ser svær ud, men den er faktisk meget enkel. Den udskriver en besked på skærmen, der lyder således: »Du gættede tallet i 3. forsøg«. Vi har gemt antallet af gæt i variablen »forsoeg«. Det er denne variabel, som her udskrives på skærmen. Som før bruger vi vores »Label«-felt til at gemme værdien i. Det interessante er det, der står efter »:=«.

```
'Du gættede tallet i ' + IntToStr(forsoeg) + '. forsøg!';
```

Igen handler det om at konvertere variabler. Denne gang skal vi bare fra et heltal og så til tekst. Funktionen »IntToStr()« gør præcis det samme som »StrToInt()« – bare omvendt. Den returnerer et tal som tekst. Dette tal indsættes så i vores tekststreng ved at bruge '+'-tegnet. Det er faktisk mere enkelt, end det ser ud, så vær ikke bange for at eksperimentere lidt.

kode.pas

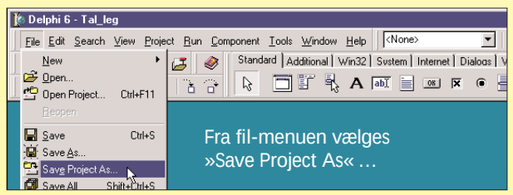
```
procedure TForm1.BtnGaetClick(Sender: TObject);
var
   gaet :Integer;

begin
   gaet := StrToInt(EdtGaet.Text);
   forsoeg := forsoeg + 1;

   if gaet = tilfaeldig then
            LblBesked.Caption := 'Du gættede tallet i ' +
                                 IntToStr(forsoeg) + '. forsøg!';

   if gaet < tilfaeldig then
      LblBesked.Caption := 'Tallet ' + IntToStr(gaet) + ' er mindre end "X"';

   if gaet > tilfaeldig then
      LblBesked.Caption := 'Tallet ' + IntToStr(gaet) + ' er større end "X"';
end;
```

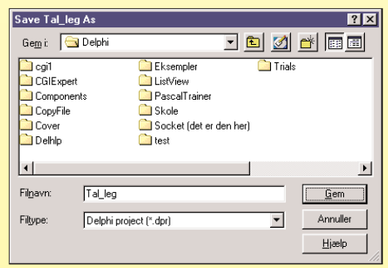
3

47: 1 Insert

GEM DIT PROGRAM

7 Nu er programmet færdigt. Det er dog altid en god idé at gemme det, før du prøver. Det gøres således:



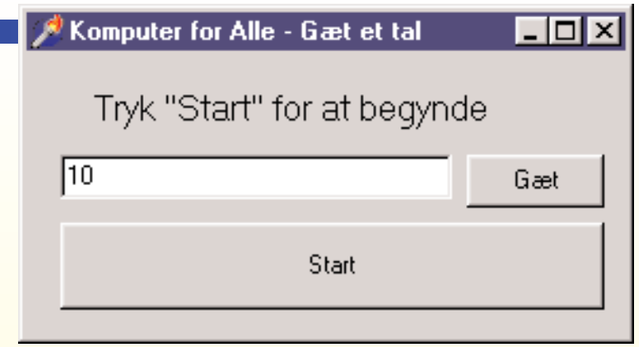


Nu beder Delphi dig om to filnavne. Det første bliver selve kode-filen gemt under, og det andet er projekt-filen.

TEST PROGRAMMET

8 Når programmet er gemt i sikkerhed på harddisken, kan du afprøve, om det fungerer korrekt.

For at starte dit program trykker du »Play«-knappen i toppen af skærmen. Hvis alt fungerer, som det skal, kan du ønske dig selv tillykke. Du har netop lavet dit første program!



9 Herfra er det op til dig at eksperimentere videre på egen hånd. Du kan for eksempel starte med at forbedre vores lille program. Eller du kan begynde på en frisk og lave dine helt egne programmer. I virkeligheden er det kun dig, der sætter grænsen for, hvad du kan lave. Og husk – Bill Gates startede også i det små!