Projekt - Visual Basic for Applications
N på stribe

Mikkel Kaas og Troels Henriksen - 03x

3. november 2005

1 Introduktion

Spillet tager udgangspunkt i det gamle kendte 4 på stribe, dog med den ændring, at brugeren selv kan sætte hvor mange (N) brikker der skal på stribe, før man har vundet. (Deraf navnet "N på stribe")
Der skal bla. laves funktioner, der tjekker, hvor der er lagt "coins"i, når en spiller vælger at lægge en coin i en given kolonne (Dette varetages af funktionen `DropCoinInColumn`). Der skal også tjekkes, om der er N brikker på stribe. (Dette varetages af funktionen `PlayerHasWon`)

2 Brugervejledning

Når *N på stribe* starter, vil det præsentere brugeren for en indstillings-dialog, hvor brugeren kan ændre navnene på de to spillere og sætte `N` (bemærk at man bør have $N > 3$, for at spillet skal være fair). Når dette er gjort, vil brugeren blive præsenteret for selve spilformen, med knapper og informativ tekst øverst, og et spilområde nederst. Ved at trykke på de knapper der er placeret over de respektive kolonner, kan spilleren placere en brik i kolonnen. Hver gang en spiller succesfuldt placerer en brik, vil det blive den næste spillers tur. Den informative tekst til venstre for indstillingsknappen i toppen af formen, vil fortælle hvis spillers tur det er. Hvis en kolonne er fyldt, vil spilleren få en fejl hvis han forsøger at placere en brik i den, og turen vil ikke skifte. Det er værd at huske på at det ikke er muligt at ændre på indstillingerne, uden at genstarte spillet - *et tryk på indstillinger-knappen vil nulstille spillet til et tomt spilbræt!* Dette er vigtigt at huske på. *N på stribe* har ingen kunstig intelligens (AI), derfor er det nødvendigt med to spillere, for at man kan gennemføre et spil.

3 Design

N på stribe er et traditionelt opbygget VBA-program, med en initialiseringsfunktion der opsætter det miljø dets funktioner er designet til at køre i, og som ellers kun handler på kommando fra brugeren. *N på stribe*s funktioner kører kun som resultat af events fra brugerens side af.

3.1 Grundlæggende teknikker og begreber

Det blå område af formen, hvor selve spillet foregår, kaldes for *spiltabellen*. Spillerne placerer *brikker* i spiltabelle, ved at vælge den *kolonne* hvor brikken skal placeres. Hver brik fylder én *celle* i tabellen.

Cellerne er implementeret som controls af typen `Picture`, og er navngivnet på formen `Cell_`*`X`*`_`*`Y`*, hvor X og Y er cellens koordinater, og hvor `Cell_0_0` ligger nederst i venstre hjørne. Når der i *N på stribe* er behov for at tilgå disse objekter bruges funktionen `Controls`. Denne tager ét parameter - en streng, der bør være navnet på en control. Controllen med det navn som man angiver i denne streng bliver returneret af funktionen, og kan så manipuleres som ethver andet objekt. Dette gør det lettere at arbejde med de i alt 100 `Picture`-objekter som spiltabellen udgør.

*N på stribe*s loops indeholder ofte et kald til `DoEvents`, der bliver udført for hver gennemkørsel (iteration) af loopet. `DoEvents` er en integreret funktion i VBA, der udfører alle event-funktioner der "står i kø." Dette indbefatter både de subrutiner der er forbundet til de forskellige controls events, men også mere low-level events, som f.eks. den funktion der tegner formen på skærmen. Dette er gjort for at programmet skal være responsivt overfor events, selv hvis man kører et loop der tager lang tid, eller som grundet en fejl udfører et uendeligt loop. Førstnævnte situation opstår især hvis brugeren sætter tyngdekraften så lavt, at det tager adskillige minutter for en brik at falde på plads. Hvis brugeren ikke har lyst til at vente på dette, kan han afslutte programmet. Idet `DoEvents` også kan forårsage *race conditions*, bruger-events der bliver kørt, når programmet ikke er i en tilstand, hvor de kan behandles forsvarligt, er det ikke alle loops der indeholder et kald til funktionen.

3.2 Overordnet programflow

De 10 pile-knapper der fungerer som spillerens primære redskab til manipulation af spillet, kalder alle funktionen `DropCoinInColumn`, med et parameter der svarer til den kolonne knappen er ansvarlig for. `DropCoinInColumn` er så ansvarlig for at checke om det er muligt at placere en brik i denne kolonne (inklusive animation), checke om det resulterer i sejr for spilleren, checke om der ikke længere er mulighed for at vinde, skifte spiller, informere spilleren om eventuelt sejr og, muligvis, starte spillet forfra. Idet denne funktion er så central for spillet, kan man sige at hele

den interne logik stammer fra et grundlæggende princip om at placere brikker i kolonner - en nøjagtig afspejling af virkelighedens 4 på stribe-spil.

4 Programbeskrivelse

N på stribe består af en række systemer, der sammen udfører en isoleret opgave. Nogle af disse systemer består kun af én funktion, andre er hele sæt af funktioner. Alle benytter de sig af en række centrale globale variabler, der bestemmer spillets tilstand.

`CurrentPlayer:` En variabel der indeholder et identifikationsnummer, der repræsenterer den aktive spiller. Det identifikationsnummersystem der bruges igennem hele *N på stribe* fungerer således:

0. Ingen spiller/blank (bruges i celler).
1. Spiller 1.
2. Spiller 2.

`GameBoardArray(9, 9):` Et array der indeholder spiltabellen. Hver celle i `GameBoardArray` er et spiller-identifikationsnummer, der viser hvilken spiller (hvis nogen) der har en brik i den celle.

`N:` Det antal brikker der skal ligge på stribe, før en spiller har vundet.

`DropInProgress:` En boolesk variabel, der indeholder information om hvorvidt en brik-animation kører.

`Gravity:` Tyngdekraften - afgør hvor hurtigt brik-animationen kører.

`LoadingLevel:` Graden af "loading" - en værdi på 0 betyder "ingen loading."

`Player1name:` Spiller 1's navn.

`Player2name:` Spiller 2's navn.

4.1 Læg brik i spil

4.1.1 DropCoinInColumn

Funktionen `DropCoinInColumn` bliver ekskveret hver gang brugeren klikker på en af pile-knapperne. `DropCoinInColumn` sørger for at der bliver tjekket hvor den nederste ledige celle i kolonnen er og at den bliver tildelt den bruger, hvis tur det er.

Parametre Denne funktion tager parametren `ColumnIndex`, der angiver hvilken kolonne spilleren ønsker at lægge sin coin i.

Returværdi `DropCoinInColumn`s returværdi er `False`

Sideeffekter Funktionen forsøger at placere en brik i kolonnen. Hvis dette kan lade sig gøre, checkes for sejr og lignende. Hvis ikke, meldes der fejl. Denne funktion sørger også for at turen skifter, hvis en brik kan placeres. Hvis brugeren har vundet, bliver vedkommende informeret om dette, og

```
PlaySoundFile ("c:\WINDOWS\Media\Tada.wav")
```

bliver kørt. Den sørger for at lydfilen Tada.wav bliver kørt vha. `winmm.dll`. Hvis spilbrættet er fyldt, melder funktionen at ingen vinder kan findes, og genstarter spillet.

Implementation For at forhindre, at der kan smides en brik i, inden den tidligere brik er faldet på plads, har vi indført boolean-variablen `DropInProgress` og lagt hele `DropCoinInColumn`-funktionen ind i en if-sætning, der afhænger af om `DropInProgress` er `True` eller `False`
Hvis `DropInProgress` er `True`, så bliver hele koden til `DropCoinInColumn` kørt. Hvis `DropInProgress` er `False`, så bliver der ikke kørt noget noget

4.2 Check for sejr

Et af de vigtigste undersystemer i *N på stribe*, er det system der holder øje med hvornår en spiller har vundet spillet. Dette system skal for det første give et korrekt resultat for enhver mulig vindende kombination, og for det andet køre hurtigt, idet det aktiveres hver gang en spiller har lagt en brik. Implementationen af dette system i *N på stribe* opfylder disse krav. Idet dette system er så kritisk for hele spillet, har vi valgt at gennemgå det i detaljer.

Meget af arbejdet med at finde ud af om en vindende stribe er til stede, går ud på at gå spiltabellen igennem på kryds og tværs, og tælle konsekutive brikker lagt af samme spiller. Den umiddelbare implementation af en sådan proces ville være at bruge en af VBA's indbyggede loop-konstruktioner, men i *N på stribe* er der i stedet brugt rekursive funktioner til meget af arbejdet. De rekursive funktioner er lettere at have med at gøre end det store antal loops, der ellers ville være nødvendigt.

4.2.1 PlayerHasWon

Funktionen `PlayerHasWon` er indgangsfunktionen til det system af funktionskald, der afgør om en given spiller har vundet, og er også den funktion der fremhæver eventuelle vindende striber af brikker.

Parametre Funktionen `PlayerHasWon` tager ét parameter, `PlayerID`, der bør være ID-nummeret for den spiller hvis vindersituation skal findes. I *N på stribe* bruges kun én værdi til dette parameter, værdien af variablen `CurrentPlayer`, så den fleksibilitet parameteriseringen giver, bliver ikke direkte udnyttet.

Returværdi Funktionen `PlayerHasWon` returnerer den booleske værdi `true` hvis spilleren angivet i `PlayerID`-parameteren har vundet, og `false` hvis dette ikke skulle være tilfældet.

Sideeffekter Hvis spilleren angivet i `PlayerID`-parameteren har vundet, vil de vindende striber automatisk blive fremhævet af `PlayerHasWon`-funktionen.

Implementation Kernen i implementationen af `PlayerHasWon` er et `While`-loop der looper over alle celler i spiltabellen (ved at iterere igennem alle gyldige værdier for `Row` og `Column`). I dette loop bliver det undersøgt om den relevante celle er en del af en vindende stribe (via funktionen `CountCoinsInSequenceIterator`. Den nøjagtige virkemåde af den funktion, vil blive gennemgået senere). Hvis dette er tilfældet, vil `PlayerHasWon` returnere `true`. Hvis ingen celler er en del af en vindende stribe vil `PlayerHasWon` returnere `false` - standardværdien, hvis intet andet sættes. I det centrale loop kaldes funktionen `DoEvents` før hver ny iteration, for at sikre at interfacet vil være responsivt, mens funktionen kører.

4.2.2 CountCoinsInSequenceIterator

Funktionen `CountCoinsInSequenceIterator` står for at finde ud af hvor mange brikker lagt af en given spiller, der ligger på en defineret række. Hvis der er nok af dem på række, vil `CountCoinsInSequenceIterator` også fremhæve dem i en gul farve, så spillerne kan se hvor der ligger *N* på stribe.

Parametre Denne funktion tager følgende parametre:

`Column:` Kolonnen for den celle der skal undersøges.

`Row:` Rækken for den celle der skal undersøges.

`ColumnVel:` Ændringen i kolonnen når funktionen kalder sig selv.

`RowVel:` Ændringen i rækken når funktionen kalder sig selv.

`PlayerID:` ID'et for den spiller, hvis brikker der skal checkes for.

`Count:` Antallet af brikker der indtil videre har ligget på række.

`N:` Det maksimale antal brikker på række der skal findes. Denne værdi kunne findes i den globale variabel `N`, men er parameteriseret i funktionen, for at sikre større fleksibilitet (der dog ikke udnyttes).

Det er `ColumnVel` og `RowVel` der definerer den række `CountCoinsInSequenceIterator` checker. For hver gang funktionen skal finde en ny brik at analysere, bliver disse to værdier lagt til de gamle værdier af hhv. `Column` og `Row`, for at danne de nye værdier, til det rekursive kald. For eksempel, hvis `Column` og `Row` begge har værdien `1`, vil `CountCoinsInSequenceIterator` tælle antallet af brikker i en diagonal linje, opad mod højre.

Returværdi Funktionen returnerer det antal brikker, lagt af den spiller `PlayerID` repræsenterer, der ligger på den række, der er angivet i parametrene. Denne returværdi kan aldrig være større end `N`.

Sideeffekter Hvis spilleren angivet i `PlayerID`-parameteren har vundet, vil de vindende striber automatisk blive fremhævet af `CountCoinsInSequenceIterator`-funktionen.

Implementation `CountCoinsInSequenceIterator` er en iterativ algoritme implementeret som en rekursiv funktion [1]. Dens virkemåde er overraskende simpel, og meget traditionel for rekursive funktioner. Når funktionen bliver kaldt, checker den parametrene, og udfører derefter en af fire handlingssekvenser - kun én af handlingssekvenserne vil blive kørt, og deres betingelser bliver checket i beskrevne rækkefølge:

1. Hvis `Count` er lig `N` betyder det at at `CountCoinsInSequenceIterator` har talt det antal brikker på række, som skal til for at striben er vindende. Hvis dette skulle være tilfældet, vil funktionen `HighlightNCoins` blive kaldt, og "sendt af sted" i en retning der er stik modsat den `CountCoinsInSequenceIterator` har. `HighlightNCoins` vil blive kaldt med parametre, der får funktionen til at fremhæve `N` brikker - altså hele den vindende stribe. Derefter vil funktionen returnere `Count`, det vindende antal.

2. Hvis `CountCoinsInSequenceIterator` har kørt så mange gange, at den er kommet uden for spilfeltet (række- eller kolonne-variabler der er enten større end 9 eller mindre end 0), vil funktionen returnere det antal brikker der indtil videre er blevet talt, den værdi der gemmes i `Count`.

3. Hvis cellen, hvis koordinater er angivet i `Column` og `Row`, indeholder en brik fra spilleren repræsenteret ved `PlayerID`, vil funktionen kalde sig selv (rekursere), med samme parametre som den selv er blevet kaldt med, med følgende undtagelser:

    - `Column` vil være ændret som bestemt af `ColumnVel`.

[1] `http://sigkill.dk/blog/archives/105-Iterative-versus-rekursive-udregningsprocesser.html`

- `Row` vil være ændret som bestemt af `RowVel`.
- `Count` vil være inkrementeret med `1`.

I dette tilfælde vil returværdien fra det rekursive kald blive returneret.

4. Hvis ingen af ovennævnte betingelser kan opfyldes, returnerer funktionen blot med værdien i parameteren `Count`.

Disse betingelser betyder at funktionen for en given celle vil returnere med det samme, med mindre cellen indeholder en brik lagt af spilleren repræsenteret ved `PlayerID`-parameteren. Værdien af `Count`-parameteren skal være `0` ved første kald til funktionen (som det ses i `PlayerHasWon`). For at undersøge enhver mulighed en given celle har for at være en del af en vindende stribe, er det nødvendigt at kalde `CountCoinsInSequenceIterator` fire gange for den pågældende celle, med forskellige værdier for `ColumnVel` og `RowVel`. Disse værdier er:

1, 0 for en linje der går lige mod højre.

0,1 for en linje der går lodret opad.

1, -1 for en linje der går nedad, mod højre hjørne.

1, 1 for en linje der går opad, mod højre hjørne.

4.2.3 HighlightNCoins

Funktionen `HighlightNCoins` bruges til at fremhæve de brikker der er en del af en vindende stribe. `HighlightNCoins` minder meget om `CountCoinsInSequenceIterator`, og er på samme måde en iterativ algoritme, implementeret som en rekursiv funktion.

Parametre

`Column:` Kolonnen for den celle der skal fremhæves.

`Row:` Rækken for den celle der skal fremhæves.

`ColumnVel:` Ændringen i kolonnen når funktionen kalder sig selv.

`RowVel:` Ændringen i rækken når funktionen kalder sig selv.

`Count:` Antallet af brikker der indtil videre er blevet fremhævet.

`N:` Det antal brikker der skal fremhæves.

Returværdi Funktionen returnerer det antal brikker der er blevet fremhævet (altid lig med `N`-parameteren).

Sideeffekter De celler som funktionen itererer over, vil blive visuelt fremhævet.

Implementation Når funktionen bliver kaldt, checker den parametrene, og udfører derefter en af fire handlingssekvenser - kun én af handlingssekvenserne vil blive kørt, og deres betingelser bliver checket i beskrevne rækkefølge:

1. Hvis `Count` er lig `N`, vil funktionen returnere `Count`.
2. Hvis ovennævnte betingelse ikke er opfyldt, vil cellen, hvis koordinater specificeres af `Column` og `Row`, blive fremhævet (ved at finde objektet via `Controls`-funktionen, og derefter sætte `Picture`-egenskaben på objektet).

4.3 Settingsform

I settingsformen bliver alle indstillinger for spillet sat. Brugeren har mulighed for at sætte N, tyngdekraften etc.
Settingsformen bliver kaldt fra spillet via `AskForSettings`

5 Tests

N på stribe's features er blevet afprøvet, og resultaterne er som følger:

Feature	*Testresultat*
3 på stribe (horisontalt, vertikalt og diagonalt)	Virker
4 på stribe (horisontalt, vertikalt og diagonalt)	Virker
5 på stribe (horisontalt, vertikalt og diagonalt)	Virker
6 på stribe (horisontalt, vertikalt og diagonalt)	Virker
7 på stribe (horisontalt, vertikalt og diagonalt)	Virker
8 på stribe (horisontalt, vertikalt og diagonalt)	Virker
Fremhævelse af vindende stribe(r)	Virker
Ændring af tyngdekraften	Virker
"Afslut"-knappen	Virker ikke
Ændring af spillernavn	Virker
Blokering for nye brikker, under brikanimation	Virker

6 Konklusion

Spillet *N på stribe* er en velfungerende implementation og generalisering, af det almindelige *4 på stribe*-spil. Det kører for det meste hurtigt (det kan dog godt mærkes, når spiltabellen begynder at blive fyldt), og spillerne har ingen mulighed for at snyde. Den eneste detalje, der måske er uhensigtsmæssig, er, at det ikke er muligt at ændre på indstillingerne under et spil.

N på stribe er generelt et solidt, vellykket spil, uden større problemer. Der er plads til optimering af algoritmerne, og en bedre udviklingsplatfom end VBA kunne være ønsket, men overordnet er projektets resultat positivt.