TDDD11 - Projekt

Grafik och tangentbordshantering

Eric Elfving

Institutionen för datavetenskap

Avdelningen för Programvara och system

LiU EXPANDING REALITY

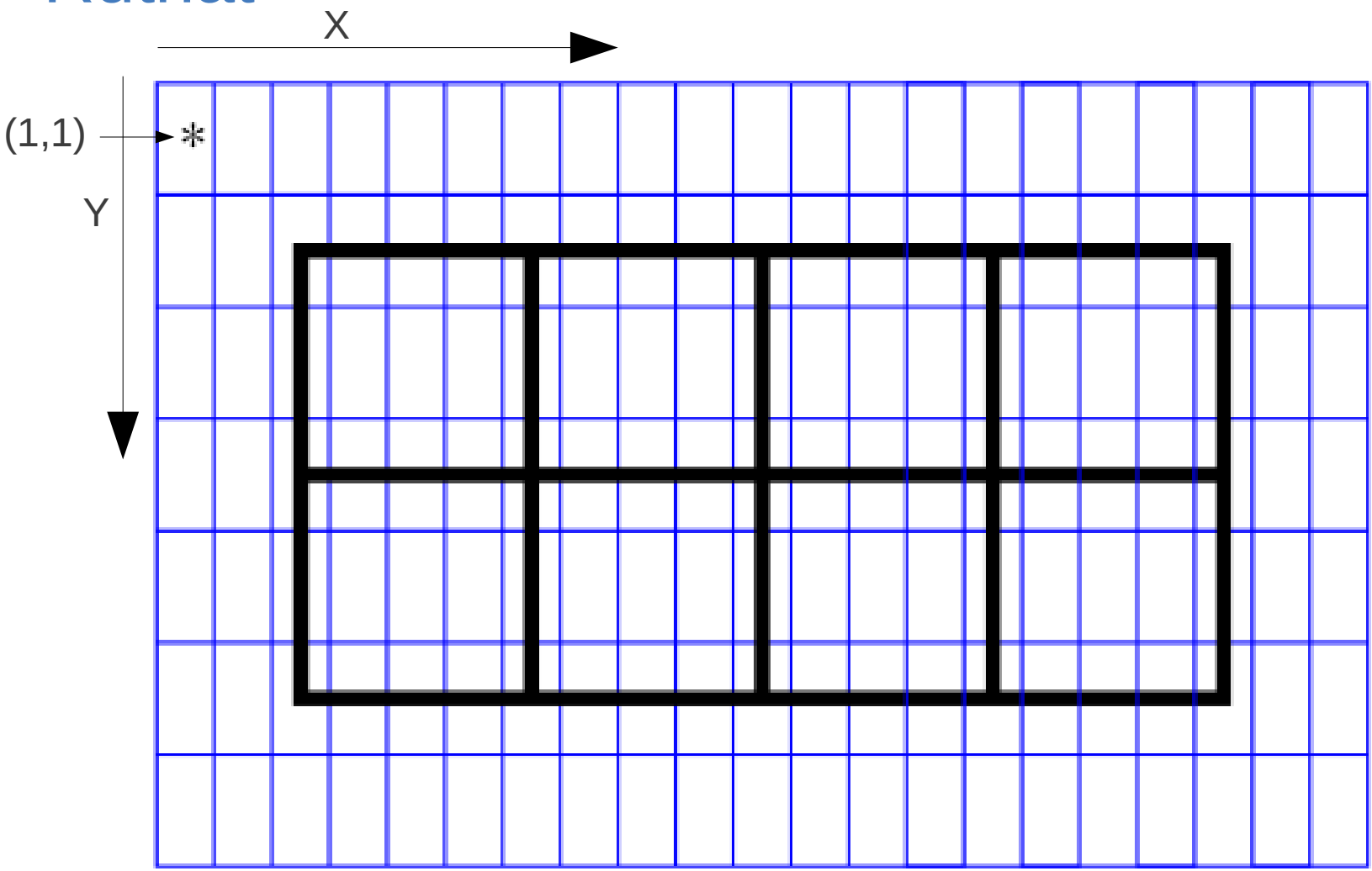
Grafik i TJa-biblioteket

- Tre nya paket:
  - `TJa.Window.Elementary` : Innehåller två procedurer, `Clear_Window` och `Goto_XY`
  - `TJa.Window.Graphic` : Hanterar grafiska tecken
  - `TJa.Window.Text` : Hanterar färger och teckenmodifieringar

Typen Graphical_Character_Type

Lower_Right_Corner	┘
Upper_Right_Corner	┐
Upper_Left_Corner	┌
Lower_Left_Corner	└
Cross	┼
Horisontal_Line	─
Vertical_Right	├
Vertical_Left	┤
Horisontal_Up	┴
Horisontal_Down	┬
Vertical_Line	│

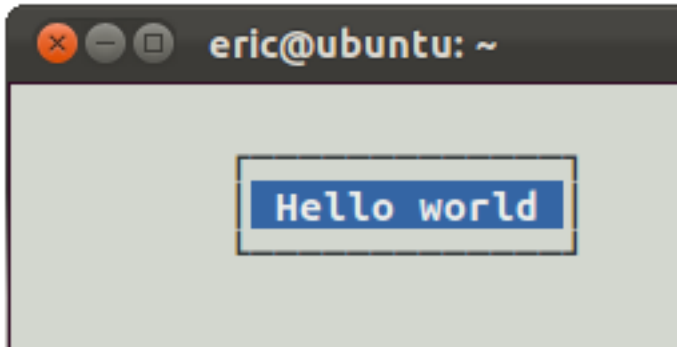
Rutnät



Tips till utritning

- Utseendet bör vara lätt att ändra!
  - Använd konstanter för diverse storlekar istället för heltal i loopar osv.
    - `No_Of_Squares_X` / `No_Of_Squares_Y` : Hur många rutor
    - `No_Of_Spaces_X` / `No_Of_Spaces_Y` : Hur stora rutor
    - `Start_X` / `Start_Y` : Vart finns övre vänstra hörnet
- Använd endast `Goto_XY`! Blir väldigt jobbigt att blanda med `New_Line`
- Kan vara bra att ha en variabel som håller koll på vilken rad vi jobbar på. Öka upp den när ni vill byta rad och använd den som Y i alla `Goto_XY`
- Som vanligt: Dela upp problemet med underprogram!

test_window.adb




```
with Ada.Text_IO;              use Ada.Text_IO;
with TJa.Window.Elementary;    use TJa.Window.Elementary;
with TJa.Window.Text;          use TJa.Window.Text;
with TJa.Window.Graphic;       use TJa.Window.Graphic;

procedure Test_Window is
   Message : constant String := " Hello world ";
   X_Start : constant Integer := 10;
   Y_Start : constant Integer := 2;
begin
   Reset_Colours;
   Clear_Window;
   Set_Graphical_Mode(On);

   Goto_XY(X_Start, Y_Start);
   Put(Upper_Left_Corner);
   Put(Horisontal_Line, Times => Message'Length);
   Put(Upper_Right_Corner);
   Goto_XY(X_Start, Y_Start + 1);
   Put(Vertical_Line);
   Goto_XY(X_Start + Message'Length + 1, Y_Start + 1);
   Put(Vertical_Line);
   Goto_XY(X_Start, Y_Start + 2);
   Put(Lower_Left_Corner);
   Put(Horisontal_Line, Times => Message'Length);
   Put(Lower_Right_Corner);
   Set_Graphical_Mode(Off);

   Goto_XY(X_Start + 1, Y_Start + 1);
   Set_Background_Colour(Blue);
   Set_Foreground_Colour(White);
   Set_Bold_Mode(On);
   Put(Message);
   Reset_Colours;
   Reset_Text_Modes;
   Goto_XY(1, Y_Start + 4);
end Test_Window;
```

Tangentbordshantering med TJa-biblioteket

- Baseras på en ny typ, **`Key_Type`**, som finns i `TJa.Keyboard`
- Tidigare har vi läst in tecken till en **`Character`** men nu kan vi läsa in en tangenttryckning till en `Key_Type`
- För att ta reda på vad för tangent som användaren tryckt på finns det ett flertal funktioner specificerade i `TJa.Keyboard`, t.ex. följande för kontroll av piltangenter:

```
-- "Is_Arrow" is the same thing as all "Is_Up_Arrow", "Is_Down_Arrow", etc.
function Is_Up_Arrow(Item : in Key_Type) return Boolean;
function Is_Down_Arrow(Item : in Key_Type) return Boolean;
function Is_Left_Arrow(Item : in Key_Type) return Boolean;
function Is_Right_Arrow(Item : in Key_Type) return Boolean;
```

Buffer_Mode / Echo_Mode

- Vanligtvis sker följande när vi trycker på en tangent med fokus i terminalen:
  - Tecknet som matchar tangenten sparas i en buffert av operativsystemet
  - Tecknet skrivs även ut i terminalfönstret
  - När användaren bekräftar inmatningen får vårt program läsa från bufferten
- Detta är dock inte så bra när vi vill skapa ett spel
  - Vi vill veta vad användaren trycker på direkt
  - Vi vill bestämma helt själva vad som ska skrivas ut
- Därför kan vi med TJa-biblioteket stänga av denna funktionalitet

Buffer_Mode / Echo_Mode

- **`Set_Buffer_Mode(Off)`**

  Inaktiverar operativsystemets buffert och vi får hantera **alla** tangentryckningar (även viktiga kommandon som Ctrl-C inaktiveras...)

- **`Set_Echo_Mode(Off)`**

  Inaktiverar utskrift av tangentbordsinmatning i terminalen

- Det är väldigt viktigt att återställa dessa i slutet av sitt program för att återgå till normala terminalegenskaper

Specialtangenter

- För att viss specialtangenter (t.ex. piltangenter) ska fungera måste man ha filen **key_codes** i den mappen man kör sitt program i. Denna fil finns bland beskrivningen av TJa-biblioteket.
- Det finns en bra procedur **`Get_Immediate`** som läser en tangent utan konfirmation (t.ex. genom att trycka på enter efter inmatning)

test_keyboard_simple.adb

(exempelkörning)

test_keyboard_simple.adb

```
with Ada.Text_IO;              use Ada.Text_IO;
with Ada.Integer_Text_IO;      use Ada.Integer_Text_IO;
with TJa.Window.Elementary;    use TJa.Window.Elementary;
with TJa.Keyboard;             use TJa.Keyboard;
with TJa.Misc;                 use TJa.Misc;

procedure Test_Keyboard_Simple is
  Key : Key_Type;
  X, Y : Integer := 10;
begin
  Clear_Window;
  Put_Line("Förflytta dig med pilarna, avsluta med ESC");
  Put_Line("Sätt ett kryss med SPACE");
  Set_Buffer_Mode(Off);
  Set_Echo_Mode(Off);
  loop
    Goto_XY(1, 20);
    Put("Current position: (");
    Put(X, Width => 2);
    Put(", ");
    Put(Y, Width => 2);
    Put(")");
    Goto_XY(X, Y);
    Get_Immediate(Key);
    exit when Is_Esc(Key);
    if Is_Character(Key) and then
       To_Character(Key) = ' ' then
      Put('X');
    elsif Is_Up_Arrow(Key) then
      Y := Integer'Max(3, Y - 1);
    elsif Is_Down_Arrow(Key) then
      Y := Integer'Min(19, Y + 1);
    elsif Is_Left_Arrow(Key) then
      X := Integer'Max(1, X - 1);
    elsif Is_Right_Arrow(Key) then
      X := Integer'Min(79, X + 1);
    else
      Beep;
    end if;
  end loop;
  Set_Echo_Mode(On);
  Set_Buffer_Mode(On);
end Test_Keyboard_Simple;
```

Allmänna tips om projektet

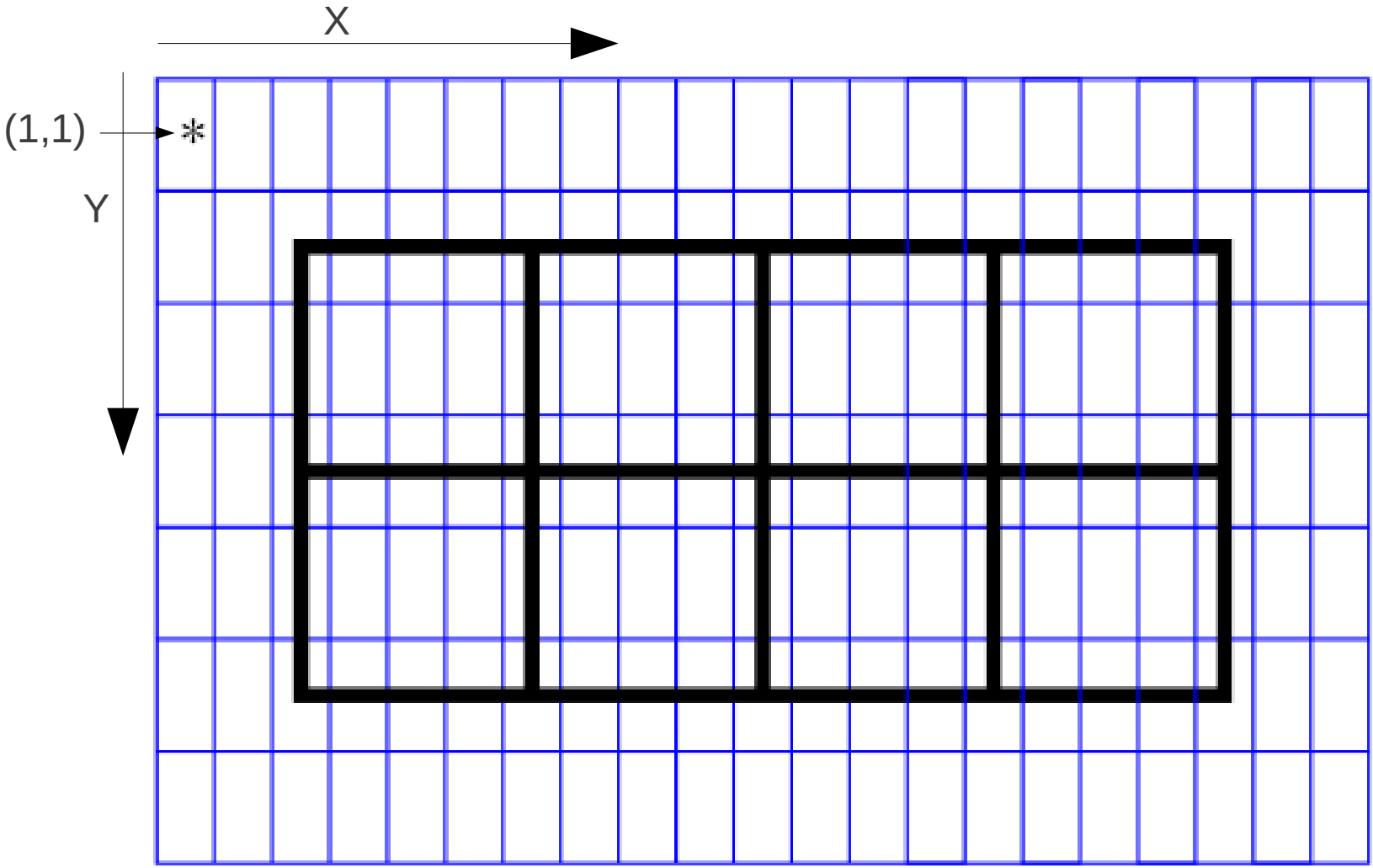
- Gör klienten till en ”dum” klient. All kontroll bör ske på server-sidan och klienten bör inte göra allt för mycket. Exempel:
  1. Server → klient: ”Hämta koordinat”
  2. Klient: Låt användaren stega omkring tills den valt position
  3. Klient → server: <koordinat>
  4. Server: Kontrollera <koordinat>
  5. Server → klient: om ok: ”Placera <markering> på <koordinat>”
     annars: återgå gå till steg 1
  6. …

Allmänna tips om projektet

- För att servern ska kunna kontrollera placering och eventuell vinst behöver den lagra information om spelplanen. Detta görs lättast i ett flerdimensionellt fält.
- Detta gör att vi får två koordinatsystem i vårt projekt, logiska koordinater och skärmkoordinater.
- Varm rekommendation: använd endast skärmkoordinater vid utritning och för att stega omkring på skärmen. Använd logiska koordinater i andra fall.
- Det kan då vara bra med funktioner för att konvertera mellan dessa.

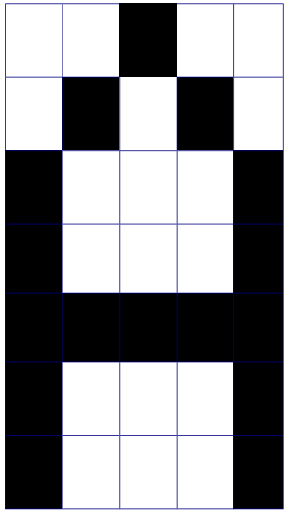
```
function To_Screen_X(X : in Integer) return Integer;
function To_Screen_Y(Y : in Integer) return Integer;

Goto_XY(To_Screen_X(X), To_Screen_Y(Y));
```

Rutnät



Allmänna tips om projektet

- För att få snyggare utritning:
  - Man kan minska teckenstorleken i terminalen och därefter skapa en bild av flera små tecken. Om jag vill skapa tecknet **A** kan det ritas ut enligt bild nedan. En ruta motsvarar här ett tecken
  - För att göra detta kan man rita ut tecknet mellanslag med en väl vald bakgrundsfärg



Resten av perioden

- Inga fler föreläsningar
- Ett till labpass med assistent (i eftermiddag och imorgon)
- Ingen lektion (”ledig” på torsdag)
- Endast projektpass utan assistent inbokade
- Om ni behöver hjälp:
  - Ställ frågor till er assistent eller boka en tid för möte via e-post (mötet sker med fördel under ett projektpass)
- Redovisning:
  - När ni känner er klara bokar ni tid för redovisning med er assistent. Detta ska ske senast det sista projektpasset, onsdagen den 9/5

Lycka till!

Linköping University

expanding reality

www.liu.se