End-User Programming of Ubicomp in the Home

Nicolai Marquardt
701.81 – Domestic Computing
University of Calgary
Outline

Introduction and Motivation

End-User Programming Strategies

Programming Ubicomp in the Home

Discussion and Summary
Introduction and Motivation
Motivation
Motivation

private void DrawBackground(Graphics g)
{
    using (Brush backBrush = new SolidBrush(BackColor))
    using (Pen borderPen = new Pen(BorderColor, 4))
    {
        g.FillRectangle(backBrush, new Rectangle(Location.X + 4, Location.Y + 4, Size.Width - 8, Size.Height - 8));
        g.DrawRectangle(borderPen, new Rectangle(Location, Size));
    }
}

/// <summary>
/// Draws the title of the item with the given index.
/// </summary>
private void DrawItemTitle(Graphics g, int index)
{
    // Set formatting and layout
    StringFormat stringFormat = new StringFormat(StringFormat.GenericNoFixedLine);  
    stringFormat.Trimming = StringTrimming.EllipsisCharacter;
    Rectangle articleRect = new Rectangle(Location.X + paddingLeft, Location.Y + rowHeight + paddingBottom, Size.Width - 4 * padding, Size.Height - 4 * padding);

    // Select color and draw border if current index is selected
    Color textBrushColor = ForeColor;
    if (index == SelectedIndex)
    {
        textBrushColor = SelectedForeColor;
    }
    using (Brush backBrush = new SolidBrush(SelectedBackColor))
    {
        g.FillRectangle(backBrush, articleRect);
        g.DrawString(title, stringFormat, textBrushColor, articleRect.X + padding, articleRect.Y + paddingTop); 
    }
}
■ **End-User Programmer:**
  “People who write programs, but not as their primary job function.” [Myers, 2006]

■ **Program:**
  “A set of statements that can be submitted as a unit to some computer system and used to direct the behavior of that system.”
  [Oxford Dictionary of Computing]
Introduction

- Controlling
  - Making advanced configurations
  - Modifying existing applications
  - Adding If-Then-Conditions
  - Creating sequences and macros
  - Using high-level programming concepts
  - Developing in Turing complete programming languages
Introduction

- Controlling
- Making advanced configurations
- Modifying existing applications
- Adding If-Then-Conditions
- Creating sequences and macros
- Using high-level programming concepts
- Developing in Turing complete programming languages
Introduction

- Controlling
- Making advanced configurations
- Modifying existing applications
- Adding If-Then-Conditions
- Creating sequences and macros
- Using high-level programming concepts
- Developing in Turing complete programming languages
Introduction

- Controlling
- Making advanced configurations
- Modifying existing applications
- Adding If-Then-Conditions
- Creating sequences and macros
- Using high-level programming concepts
- Developing in Turing complete programming languages
Introduction

- Controlling
- Making advanced configurations
- Modifying existing applications
- Adding If-Then-Conditions
- Creating sequences and macros
- Using high-level programming concepts
- Developing in Turing complete programming languages
Introduction

- Controlling
- Making advanced configurations
- Modifying existing applications
- Adding If-Then-Conditions
- Creating sequences and macros
- Using high-level programming concepts
- Developing in Turing complete programming languages

Complexity and level of abstraction

Low

High

End-user programming
End-User Programming Strategies
- Make abstract and high-level programming concepts understandable

**Threshold and Ceiling:**

“The *threshold* is how difficult it is to learn how to use the system, and the *ceiling* is how much can be done using the system. “

[Myers, Hudson, Pausch, 2000]

- Low *threshold* and high *ceiling*
End-User Programming

1. Simplified programming languages
2. Visual programming systems
3. Natural language interpretation
4. Programming by demonstration/example (PBD/PBE)
Simplified Programming Languages

- Making programming languages easier to understand
- BASIC
- LOGO
Simplified Programming Languages

![Image of a programming interface with code and a question mark]
Simplified Programming Languages

```
<table>
<thead>
<tr>
<th>PEN DOWN</th>
</tr>
</thead>
<tbody>
<tr>
<td>REPEAT 9</td>
</tr>
<tr>
<td>LEFT 90</td>
</tr>
<tr>
<td>REPEAT 3</td>
</tr>
<tr>
<td>SET COLOUR Blue</td>
</tr>
<tr>
<td>FORWARDS 100</td>
</tr>
<tr>
<td>LEFT 20</td>
</tr>
<tr>
<td>END REPEAT</td>
</tr>
<tr>
<td>LEFT 120</td>
</tr>
<tr>
<td>REPEAT 3</td>
</tr>
<tr>
<td>SET COLOUR Red</td>
</tr>
<tr>
<td>FORWARDS 100</td>
</tr>
<tr>
<td>LEFT 20</td>
</tr>
<tr>
<td>END REPEAT</td>
</tr>
<tr>
<td>END REPEAT</td>
</tr>
<tr>
<td>PEN UP</td>
</tr>
</tbody>
</table>
```

The program draws a flower shape on the screen.
Visual Programming

[Myers, 1986]: William Sutherland, 1966 – Graphical Programming
Visual Programming

Interpolation
- Start Value
- End Value
- Duration
- Tension
- Repeat Mode
- Interpolation

Image With String
- String
- Font Name
- Font Size
- Font Loading Offset
- Font Kerning Shift
- Image Width
- Image Height

Sprite
- Enable
- X Position
- Y Position
- Z Position
- X Rotation
- Y Rotation
- Z Rotation
- Width
- Height
- Caller
- Image
- Mask Image
- Blending
- DepthTesting
- Face Culling
Visual Programming

Problems

Enable the "world" macro once the text for the RSS feed has been downloaded.

Boolean Splitter
- Input
- Output

Sample & Hold
- Sample Value
- Value
- Sampling
- Reset Signal

RSS Downloader
- URL
- Update Signal
- Article List
- Download Progress
- Done Signal

getNextIndex
- articleList
- Output
- next

RSS Image Downloader
- articleImage
- imageDoneSignal
- imageSize
- Download Progress

World
- Enable
- Image
- Start
- Time

Composition Loader
- Enable
- Composition Location
- _protocolInput_PrimaryColor
- _protocolInput_SecondaryColor
- position

Particle System
- Enable
- Particle
- Count
- X Position
- Y Position
- Color
- X Min Velocity
- X Max Velocity
- Y Min Velocity
- Y Max Velocity
- Size
- Lifetime
- Attraction
- Gravity
- Image
- Blending
- Color
- Size Delta
- Red Delta
- Green Delta
- Blue Delta
- Opacity Delta

LFO
- Type
- Result
- Period
- Phase
- Amplitude
- Offset
- PWM Ratio

Blurry Texture
- Render
- Texture Target
- Texture Mipmapping
- Pixels' Wide
- Pixels' High
- Center Y
- Center X

Image With String
- String
- Font Name
- Display Width
- FontWeight
- Font Size
- Display Height
- Font Leading Offset
- Line Count
- Font Kerning Shift
- Character Count
- Image Width
- Image Height

Image Importer
- Image

3D Transformation
- Enable
- Rotation Origin X
- Rotation Origin Y
- Rotation Origin Z
- Rotation X
- Rotation Y
- Rotation Z
- Translation X
- Translation Y
- Translation Z

Billboard
- Enable
- Width
- Image
- Mask Image
- X Position
- Y Position
- Rotation
- Pixel Aligned
- Blending
- Texture
- Size

FPS Display
- Enable
- Display Width
- Display Height
- Font Name
- Font Size
- Font Leading Offset
- Line Count
- Font Kerning Shift
- Character Count
- Image Width
- Image Height
Always store the images from my digital camera online for sharing.

What should I do exactly...

- Detecting the correct camera
- Loading images to computer
- Login on web service, upload the photos
- ...

OK. I will automatically upload photos to Flickr if your digital camera is connected to this computer...
Programming by Demonstration

define

drag

drop

[Lieberman, 2001]
End-User Programming

Other Strategies

Form- or template-based programming

Dialog-guided (wizard) programming

Sequence and macro recording
Programming Ubicomp in the Home: Example System Prototypes
Programming Ubicomp in the Home

CAMP – Magnetic Poetry
Programming Ubicomp in the Home

“always record pictures of baby Billy and display at my location a picture of baby Billy”
“always record pictures of baby Billy and display at my location a picture of baby Billy”
Programming Ubicomp in the Home

[Dey at al., 2004]: System a CAPpella
Programming Ubicomp in the Home

[Jigsaw Metaphor]

[Humble et al., 2003]
Programming Ubicomp in the Home

Jigsaw Metaphor

[Humble et al., 2003]
Programming Ubicomp in the Home

Tangible, Education

[Horn & Jakob, 2007]
Programming Ubicomp in the Home

[Horn & Jakob, 2007]
Programming Ubicomp in the Home

[iCAP, Form Based]

[Sohn & Dey, 2003]
Programming Ubicomp in the Home

Options

AND

Properties

OR

AND

[Sohn & Dey, 2003]
Programming Ubicomp in the Home

[Beckmann & Dey, 2003]
Programming Ubicomp in the Home

[Beckmann & Dey, 2003]
Programming Ubicomp in the Home

Magic Cubes

[Blackwell & Hague, 2001]
Discussion and Summary
Discussion

- **Low threshold** → low ceiling?
- Difficult: Making high-level programming concepts, boolean logic, and abstractions **easier to understand**
- Users think in “**functionality**”, not in “devices”
- Handling **exceptions** (overriding system decisions)
- Interactive and immediate **feedback**
- Simplified **debugging** mechanisms
- Handling **conflicts/contradictions/ambiguity**
Common Technical Challenges

- Service-Oriented Architectures (SOA)
- Recombinant computing
- Mobile code frameworks, runtime binding
- Dynamic discovery
- High fault tolerance, redundancy
Summary

Various strategies

Fundamental: low threshold

Aiming for: high ceiling

Make abstractions understandable

Users: functionality vs. devices
END-USER PROGRAMMING OF UBICOMP IN THE HOME


END-USER PROGRAMMING IN GENERAL

Using tangible interface to teach children how to simply create simple programs to control robots.

This book introduces the concepts of programming by example (or: demonstration). It also explains a wide area of research prototype systems, and discusses their advantages and limitations.

Detailed overview of visual programming and visualizations for programming. Covers many of the early systems.

Describes successful and failed approaches in the development of user interface tools. In the paper the authors also predict important paradigms for the future development of user interface tools.

Very detailed presentation about the objectives and challenges of end-user programming. Talk was given at CHI 2006 (slides are available on the referenced website).

OTHER REFERENCES


Photos and graphics: stock.xchng, or the references publications
Thank you for your attention