

您的潜力, 我们的动力

Microsoft
微软(中国)有限公司

创建能在各式设备上运行的 **Windows Mobile**应用程序

Name 胡砚 (MVP)

Job Title 技术总监

Company 北京拓杰凯业科技有限责任公司



微软中文技术论坛

您的潜力. 我们的动力

Microsoft®
微软(中国)有限公司

——精彩生活每一天

<http://forums.microsoft.com/china>

本周活动更新：

- ✓ Top 10 论坛英雄！
- ✓ 畅谈我的2007



与众不同：

- ✓ 版主：50+ 微软最有价值专家（MVP）
- ✓ 涵盖微软几乎所有产品线和知识库
- ✓ 30+ 适合开发人员和 IT 专业人员技术板块

City Event Sponsor Program

相聚同城技术培训



MSDN Webcasts

Agenda

- The Big Picture
- Device convergence
- Handling device display & form factor differences
- Adapting to other device behavior differences

The Windows Mobile Family

- Windows Mobile provides many different devices
 - Different input capabilities
 - Many different displays
 - Phone support
 - Windows Mobile version differences
 - Hardware differences

Device Differences

您的潜力. 我们的动力

Microsoft
微软(中国)有限公司

Device differences are good for us

- Differences create opportunity
 - More user choices make devices more competitive
 - Easier for users to find a device that is “right for the job”
- Differences break down into 3 categories
 - Availability of touch-screen support
 - Form factor
 - Extended capabilities
 - Windows Mobile version
 - .NET Compact Framework version
 - Available Hardware

Device Differences

How Windows Mobile & developer tools help

- Windows Mobile platform & tools
 - Win32 provides consistent APIs across devices
 - .NET Compact Framework provides consistent runtime
 - C\C++ provides DeviceResolutionAware.h
 - Visual Studio 2005
 - Wide variety of emulators

Handling Device Differences

Maximize each device experience

- Supporting device differences should be fundamental
- Incorporate features from newer platforms & devices
 - Provide better user experience
 - Avoid targeting lowest common denominator
 - Downgrade gracefully for older or more basic devices

Agenda

- The Big Picture
- Device convergence
- Handling device display & form factor differences
- Adapting to other device behavior differences

The Times They Are A-Changin'

Don't think Pocket PC vs. Smartphone

- Don't focus on Pocket PC vs. Smartphone
- Many Pocket PCs have phone support
- Smartphone screen sizes are increasing
- All Windows Mobile devices now have soft keys
- Both PPC & Smartphone may have full keyboard
- The Difference is Touch Screen

Device Convergence

Microsoft
微软(中国)有限公司

Windows Mobile 6

Convergence reflected in Windows Mobile 6 naming

Windows Mobile 6 Standard device

- Touch-screen: No, Phone: Yes
- The device formerly known as Smartphone

Windows Mobile 6 Professional device

- Touch-screen: Yes, Phone: Yes
- The device formerly known as Pocket PC Phone Edition

Windows Mobile 6 Classic device

- Touch-screen: Yes, Phone: No
- The device formerly known as Pocket PC

您的潜力. 我们的动力

Device Convergence

Microsoft
微软(中国)有限公司

Windows Mobile 6 SDKs

- New SDKs partitioned by touch-screen support

Windows Mobile 6 Standard SDK

Used for devices with no touch-screen support

Windows Mobile 6 Professional SDK

Used for devices *with* touch-screen support



— Think of as “Windows Mobile 6 SDK for

MSDN Webcasts

幻灯片 11

HL1

Can visualize by putting old name on left => new name, then image or text (sub bullet) - pointing out whether touch-screen or non touch-screen

Heather Littlejohn, 2007-3-29

Touch-Screen Support Differences

Non-touch-screen devices

Often designed for one-handed usage

Navigate by sequentially moving between controls

Menus available through soft keys

Touch-screen devices

Support everything that non-touch-screen devices support

May take advantage of two-handed usage to speed input

Support random navigation via stylus

May use context menus to simplify menu selections

Support handwriting input

Touch-Screen vs. Non-touch-Screen

Designing Portable Apps

- Limit application to features available both

Easy to develop

No Touch features

- Application auto-upgrades for touch-screen devices
 - App's basic design supports nontouch-screen
 - App “turns on” features for touch-screen

Capitalizes on Features

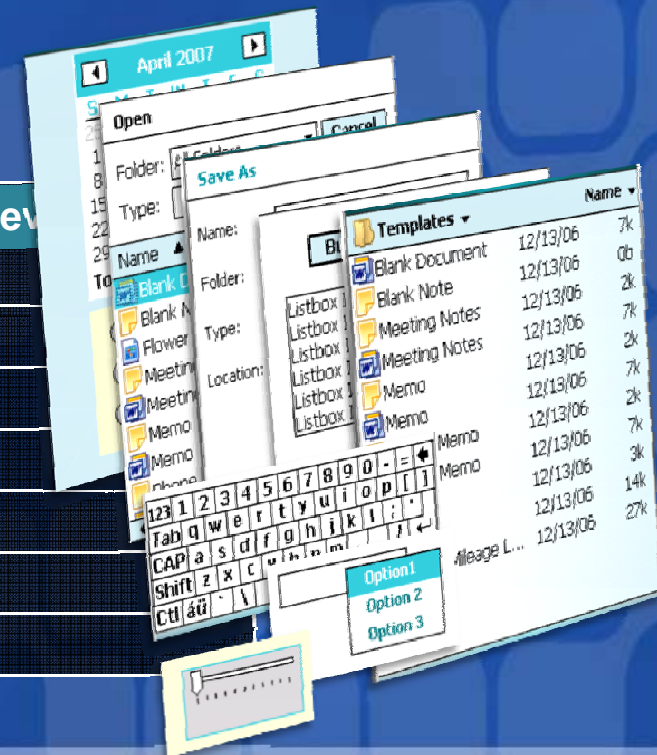
More Effort to Develop

Lowest Common Denominator

Compatibility by Focusing on Non-touch-Screen

- Limiting features to non-touch-screen devices
 - Avoid Ink input
 - Avoid unsupported UI features

.NET CF 2.0 controls not supported by nontouch-screen devices		
Button	MonthCalendar	StatusBar
ContextMenu	Notification	TabControl
DocumentList	NumericUpDown	ToolBar
DomainUpDown	OpenFileDialog	TrackBar
HardwareButton	RadioButton	
InputPanel	SaveFileDialog	
ListBox	Splitter	



HL2

List of controls. Speaker is going to send screenshots of the controls in the table below (will send tomorrow). Maybe just use a few as visual elements. Need to clean up.

Heather Littlejohn, 2007-3-29

Auto-Upgrade Applications

Enable Additional Features for Touch-Screen

- Create basic non–touch-screen layout
- Determine device touch-screen support on startup
- Create additional controls when on touch-screen
- Don't over do it
 - Focus on easier ways to perform existing features
 - Avoid introducing entirely new features
 - Be careful not to change UI too much

您的潜力. 我们的动力

Microsoft
微软(中国)有限公司

DEMO

Touch-Screen vs. Non-touch-screen

Creating an Auto-Upgrade
Application

Agenda

- The Big Picture
- Device convergence
- Handling device display & form factor differences
- Adapting to other device behavior differences

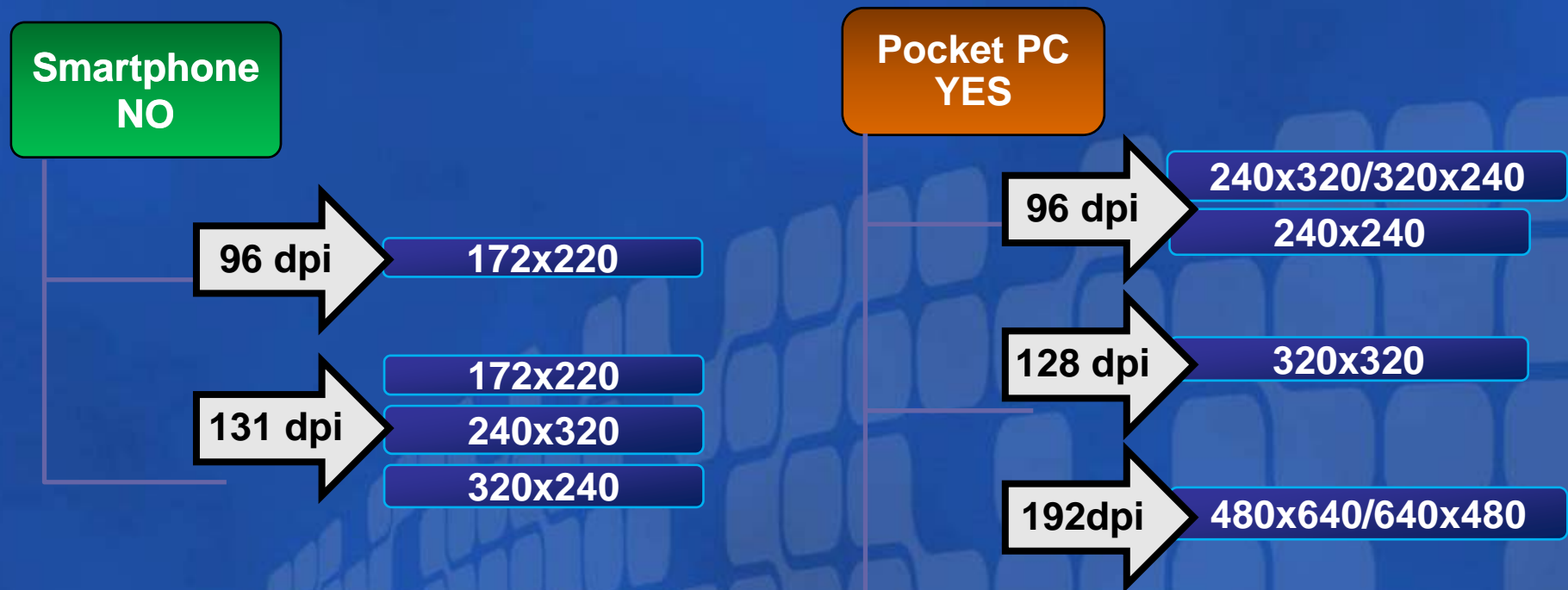
Device Form Factor Differences

Managing user interface differences

- Windows Mobile provides wide array of form factors
 - Many different sizes
 - Portrait, landscape, and square orientation
 - 96, 128, 131, 192 dots-per-inch (dpi) resolutions
- Difficult to create a one-size fits all user interface
 - User interface must adapt
 - Interrogate and adapt to display settings at start up
 - Orientation may change during application execution

Device Form Factor Differences

The Current List



- * 320x320 form factor new with Windows Mobile 6
- More form factors likely to come

Handling Form Factor Differences

- Windows Mobile 6 UILayout sample
 - Includes reusable class, ScreenLib
 - Provides functions for handling common screen layout issues
- Win32 provides over 20 functions & macros
 - Found in DeviceResolutionAware.h
- .NET CF provides display properties
 - Control.Dock
 - Control.Anchor
 - Control.AutoScroll
 - Control.AutoScale

Handling Form Factor Differences

Fine tuning

- Control class properties sometimes not enough
 - Need some way to adjust UI for form factors
 - May need to significantly reposition/resize some controls
 - Manually adjusting controls may not be most desirable choice
- Patterns & practices OrientationAwareControl

Handling Form Factor Differences

Finding the right solution: Things to avoid

- Avoid spreading layout-handling code throughout controls
 - As much as possible, keep code in app startup & Resize event
- Avoid targeting specific size, resolution, orientation combos
 - The list is large
 - New form factors coming all the time

Handling Form Factor Differences

Complex user interfaces

- May have to split UI components across different forms
- More important than ever to separate UI from logic
- Avoid getting stuck in a form-based mindset
- Try to break UI components into points of interaction
- Centralizes logic for mapping UI components to forms

Agenda

- The Big Picture
- Device convergence
- Handling device display & form factor differences
- Adapting to other device behavior differences

Creating Adaptive Apps

- Device differences not limited to display
 - Several generations of devices now exist
 - Devices becoming increasingly rich
 - Sharable resources more prolific
- Adaptive application behavior
 - Determine device characteristics
 - Provide best experience for current device
 - Centralize specialization

Device Characteristics

- Operating system version
 - Environment.OSVersion.Version
- .NET Compact Framework runtime version
 - Environment.Version
- . Does device have a phone?
 - Windows Mobile 5.0 or greater
 - SystemState.PhoneRadioPresent
 - Prior to Windows Mobile 5.0
 - File.Exists(@"\Windows\Phone.dll")

Device Characteristics

Accessing other characteristics

- State and Notifications Broker
 - Available to Windows Mobile 5.0 and greater
 - Windows Mobile 5.0: Nearly 110 states/characteristics
 - Windows Mobile 6: Nearly 150 states/characteristics
- Retrieve device configuration information
 - Managed code: ConfigurationManager
 - Native code: QueryPolicy/DMPProcessConfigXML
- Registry
 - Limit use to information not available elsewhere

Adaptive Behavior

Style guidelines

- Adaptive behavior style guidelines
 - Avoid putting “if” statements throughout code
 - “if” statements complicate maintenance
 - Tend to be inefficient—Frequently repeat same tests
 - Focus on centralizing code
 - Limit repeated tests
 - Group related code together
 - Keep adaptive code out of your app
 - Simplifies coding
 - Simplifies adding and managing behaviors

Adaptive Behavior

Implementation best practices

- Externalize adaptive code
 - Create separate assembly
 - Simplifies adding new implementations
- Use factory model
 - Provides common path for specialized instance creation
 - Centralizes decision making
 - Avoids repeated testing—simplifies debugging
- Abstract desired behavior into interfaces
 - Keeps app focus on what to do, not how to do it
 - Simplifies adding new implementations

您的潜力. 我们的动力

Microsoft
微软(中国)有限公司

DEMO

Creating Adaptive Applications

Summary


- Avoid thinking in terms of device type
- Think in terms of touch-screen support
- Design applications from the start for adaptability
- Consider adaptability a core requirement
- Separate adaptability code from core app logic

获取更多MSDN资源

- **MSDN中文网站**
<http://www.microsoft.com/china/msdn>
- **MSDN中文网络广播**
<http://www.msdnwebcast.com.cn>
- **MSDN Flash**
<http://www.microsoft.com/china/newsletter/case/msdn.aspx>
- **MSDN开发中心**
<http://www.microsoft.com/china/msdn/DeveloperCenter/default.msp>



Question & Answer

如需提出问题，请单击“提问”按钮并在随后显示的浮动面板中输入问题内容。一旦完成问题输入后，请单击“提问”按钮。

 **问题和解答 (无问题)** ▲ ×

在此会议中尚未解答任何问题。

要向演示者提问，请在此处键入问

提问(A)

删除(D)

问题管理器(Q)

您的潜力，我们的动力

Microsoft®
微软(中国)有限公司

Microsoft®

msdn


MSDN Webcasts