Today’s Lecture

- Wiimote – Example Code
  - Walk through Wii Balance Board example code
- Discussion
  - Intelligent Assistance
- Properties – Graphic Interfaces
  - Fundamentals
  - Strengths / Weaknesses

Reminders

Blog Post (Week)
- C# Programming
- Project 1

How autonomous?

South Park Episode #61: Trapper Keeper
http://www.southparkstudios.com/episodes/103902
Small Group Exercise

☐ How intelligent should our computer systems be?

Split into groups of 2-4 students

Blog Post – Week 3

☐ Writing assignment
  ■ Is time better spent on smarts (anticipating / learning) or on improving the interface?
  ■ Give one or two examples – pictures / etc.
  ■ Writeup – three to four paragraphs

☐ Receive feedback on your post
  ■ Worth 20 points

WiimoteLib – Adding a Reference

Do this for
- External DLLs
- Other Projects

DLL = Dynamic Link Library
Browse for the DLL

Use it in your project

```csharp
using WiimoteLib;
```

Look at ExampleBalance

- Posted on the wiki
- Interacts with Wii Balance Board
- Written with WPF
- Key aspects
  - Borrows heavily from the Wii Drum Kit (Brian Peek’s site)
  - Delegates
  - Dispatcher
Demo – Balance Board

Code Structure

- Constructor
  - Create controls – InitializeComponent
  - Find wiimotes – InitializeWiimote

- Delegates
  - One for extensions changing
  - One for general change

- Dispatcher
  - Change event -> dispatch
  - Thread issue with WPF

InitializeWiimote

```csharp
private void FindWiimotes()
{
    try
    {
        mWC.FindAllWiimotes();
    }
    catch (WiimoteNotFoundException ex)
    {
        // …
    }
    catch (WiimoteException ex)
    {
        Console.WriteLine("Wiimote error: " + ex.Message);
        this.Close();
    }
}
```
Review – Exception Syntax

```csharp
try
{
    // Normal code here
}
catch (ExceptionType ExceptVarName)
{
}
catch (ExceptionType ExceptVarName)
{
}
```

Exceptions

- Try something tricky
  - File I/O
  - Network I/O
- Extra guard against failure
  - One way ticket out though
  - Cannot resume back in code
- Design functions to throw exceptions
  - Ex. input argument is NULL
  - Base on assertions / pre-conditions
    - Pre / Post conditions??
    - Lambda calculus??
    - Formal methods

Example

```csharp
try
{
    // Discuss top secret plans
}
catch (SpanishInquisition ex)
{
    Console.WriteLine("Nobody expects...");
    Console.WriteLine("well, except us");
}
Delegates, Events
- Better function pointers from C/C++
  - One to one mapping
  - Multicast
    - One to many
    - Publish / subscribe
  - Why?
    - Allows us to decouple data vs. reaction
    - Ex. Clock tick
    - Ex. Button press
    - Ex. Wiimote change
  - Derive child vs. enhance

Events vs. Delegates
- Event is a simpler form of delegate
  - Source -> who did it?
  - Event arguments -> what happened?
    - Derived from base EventArgs class
    - That's it
    - Helper -> EventHandler to specify
    - No return – only void
- Delegate
  - More generic
  - Any parameters we want

Event Example - Wiimote
```csharp
// (summary)
// Event raised when Wiimote state is changed
///<summary>
public event EventHandler<WiimoteChangedEventArgs> WiimoteChanged;
```

**event** keyword
**EventHandler**
List of subscribers
Who is listening?
**Event Example – Wiimote Core Code**

```csharp
/// <summary>
/// Event raised when Wiimote state is changed
/// </summary>
public event EventHandler<RemoteEditTextEventArgs> RemoteEditTextChanged;

NAME -> Could be made into a property
```

**Example – Using events**

**What we want to get invoked**

```csharp
/// <summary>
/// Wiimote state changed handler
/// </summary>
private void wm_RemoteTextChanged(object sender, RemoteEditTextEventArgs args)
{

OBJECT -> base object
SENDER -> source

Note the same Event argument type
```

**Subscribing to the event**

```csharp
// Connect wiimotes and set them up as always
foreach (Wiimote wm in wms)
{
    wm.WiimoteChanged += wm_WiimoteChanged;
    wm.WiimoteExtensionChanged += wm_WiimoteExtensionChanged;

    The public event handler
Our subscriber

Each time the Wiimote values "change", it calls our method
```
Discuss

☐ Any GUIs that this might be useful with?

Split into groups of 2-4 students

Dispatch

Dispatcher for this window

Queue up a task to be run later

Why? Thread Safety / WPF Quirk

Dispatch

A newly created delegate

Special type just for this case

Just wraps the event handler
Delegate definition

// Delegate and text format change delegate
private delegate void UpdateViewHolderChangedDelegate(object sender, UpdateViewHolderChangedEventArgs args);

Pure type only
Key aspect is the EventArgs type

Dispatch

private void OnViewHolderChanged(object sender, UpdateViewHolderChangedEventArgs args)
{
    UpdateViewHolderChangedDelegate delegate = (UpdateViewHolderChangedDelegate)sender;
    delegate(sender, args);
}

// Standard
// Handle Section Change of Galaxy State
// Handle Section
// Galaxy Section SectionChangedDelegate SectionArgs ASPX
// GalaxySection.cs w3 = ASPX.GalaxySection:

Really?

☐ Do I have to remember all this?
  ■ General usage of delegates / events
    ☐ Yes
  ■ Dispatch / WPF quirks
    ☐ Sort of enough to get your code going

Example mini-WPF code base for you to start from in Example Code
Back to the notes

☐ GUI Pros / Cons
  ■ If time allows

Questions?

- Homework 2
- Week 2 / 3 – Blog
- Project 1