Using QuickSilver to mock-up Web 3.0

Krzys Ostrowski
Introduction
BAD

$100m big, scary and ugly server farm

event
BAD

virtual room’s “home” machine

player in the room

event
Architecture
"game server"

- content "home"
- GMS

Connections:
- enter
- join
- info
- manage
- events
Tools
Development Environment

• **Microsoft Visual Studio 2005 “Express”**
  (supports XNA Framework & Game Studio Express)
  – XNA Project types
  – Implements the “content pipeline”

• **Microsoft Visual Studio 2005 “Professional”**
  (support only the XNA Framework)
  – Version control,
  – Debugging multithreaded applications, etc.
protected override void LoadGraphicsContent(bool loadAllContent)
{
    if (loadAllContent)
    {
        // TODO: Load any ResourceManagementMode.Automatic content
        myTexture = content.Load<Texture2D>("Krzys");
        player = CreatePlayer(n, Vector2.Zero);
    }
}
Coding
How to access the functionality provided by XNA, QMS etc.
<table>
<thead>
<tr>
<th>Component Name</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsoft.VisualStudio.Web.Application</td>
<td>8.0.0.0</td>
</tr>
<tr>
<td>Microsoft.Vsa</td>
<td>8.0.0.0</td>
</tr>
<tr>
<td>Microsoft.Vsa.Vb.CodeDOMProcessor</td>
<td>8.0.0.0</td>
</tr>
<tr>
<td>Microsoft.VSSDK.UnitTestLibrary</td>
<td>8.0.0.0</td>
</tr>
<tr>
<td>Microsoft.Xna.Framework</td>
<td>1.0.0.0</td>
</tr>
<tr>
<td>Microsoft.Xna.Framework.Content.Pipeline</td>
<td>1.0.0.0</td>
</tr>
<tr>
<td>Microsoft.Xna.Framework.Game</td>
<td>1.0.0.0</td>
</tr>
<tr>
<td>Microsoft.Vsa.Vb</td>
<td>8.0.0.0</td>
</tr>
</tbody>
</table>
Add a pre-build event to manually copy QuickSilver_0.dll,
You can also copy over graphical content here
Where to write your own code
namespace MyGameClient
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
            gamethread = new Thread(new ThreadStart(ThreadCallback));
            gamethread.Start();
        }
    }
}
public partial class Form1 : Form
{
    ...

    private void ClosingCallback(
        object sender, FormClosingEventArgs e)
    {
        ...
    }
}
namespace MyGameClient
{
    static class Program
    {
        [STAThread]
        static void Main(string[] args)
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1(args[0], args[1], args[2]));
        }
    }
}
When does your code run?
Threads

- **Game Thread**
  - Rendering graphics
  - Respond to mouse/keyboard

- **Windows Forms Thread**
  - Rendering Forms
  - Respond to mouse/keyboard

- **QSM Thread (inside QSM)**
  - Network communication

- **Your own threads?**
  - Your background processing etc.

```csharp
Application.Run(new Form1(...));
game = new MyGame(...);
game.Run();
client = QS.CMS.Framework2.Client.Create(...);
```
Your Code

• Your own threads
• Callbacks from Windows Forms (GUI Thread)
  – Mouse, keyboard etc.
• Callbacks from Game (Game Thread)
  – Update(GameTime gameTime)
  – Draw(GameTime gameTime)
• Callbacks from QSM (QSM Thread)
  – When packets are received, sent, acked etc.
  – Must do minimum work and leave!
How to use QSM
Launching QSM “client”

```csharp
using QS.CMS.Framework2;

...

ClientConfiguration configuration = new ClientConfiguration(
    "192.168.0.0/24",
    10000,
    "192.168.0.100");
configuration.Verbose = true;

...

IClient client = Client.Create(configuration);
```
Joining a QSM group

```csharp
IGroup group = client.Open(
    new QS.CMS.Base3.GroupID(groupid),
    GroupOptions.FastCallbacks);
```

- how QSM interacts with your code
- numeric identifier
Receiving from a group

```c
private void ReceiveCallback(QS.CMS.Base3.InstanceID sender, QS.CMS.Base3.ISerializable message)
{
    Message m = (Message) message;
    ...
}
```

called on receive

your class
Sending to a group (1)

group.BufferedChannel.Send(new Message(...));

- uses internal buffer to store messages you are sending
- you can send this way from any thread
Sending to a group (2a)

- register to send
- feed
- sink
- get messages
- policy
  - limit sending rate
  - limit concurrency
  - limit window size

elements of the protocol stack:

- app
- f/s
- sock
- socket
Sending to a group (2b)

```csharp
private void SendCallback(
    IChannel channel,
    uint maxsend,
    out bool hasmore)
{
    channel.Send(new Message(...));
    hasmore = false;
}
```
Serialization

• Built-in XML
  – Flexible, easy to use
  – Slow
  – Space-consuming

• Built-in “binary”
  – Useless

• QSM
  – Fast, supports scatter-gather I/O
  – Takes getting used to...
“traditional” serialization

Foo

- X : int
- S : string

Packet

- size : int
- buff : char[]

X

S

size

buff

copy

copy

copy (expensive)
“scatter-gather” serialization

Foo

X : int
S : string

Packet

reference

X

reference

S

size : int
buff : char[]

reference

size

reference

buff

......
serialization in QSM

Foo

X : int
S : string

Packet

reference

header

copy

reference

size

size : int
buff : char[]

buff

......
Anatomy of a serializable class (1)

(1) [QS.CMS.Base2.ClassID(Message.ClassID)]

public sealed class Message : QS.CMS.Base3.ISerializable
{

    public const ushort ClassID = (ushort)(QS.ClassID.UserMin + 100);

    public Message()
    {
    
    
    }   (4)
}   (3)
Anatomy of a serializable class (2)

unsafe QS.CMS.Base3.SerializableInfo

QS.CMS.Base3.ISerializable.SerializableInfo

{ (5)
  get {
    get
      return
        new QS.CMS.Base3.SerializableInfo(
            ClassID, sizeof(uint) + 2 * sizeof(float)));
  }
}

class identifier

header size
Allow unsafe code
unsafe void QS.CMS.Base3.ISerializable.SerializeTo(
    ref QS.Fx.Base.ConsumableBlock header,
    ref IList<QS.Fx.Base.Block> data)
{
    fixed (byte* arrayptr = header.Array)
    {
        byte* headerptr = arrayptr + header.Offset;
        *((uint*)headerptr) = playerid;
        *((float*)(headerptr + sizeof(uint))) = x;
        *((float*)(headerptr + sizeof(uint) + sizeof(float))) = y;
    }
}

here you can copy append your buffers here
How to use WCF
reference your class lib from client and server

reference WCF DLLs

shared class library
using System.ServiceModel;

namespace MyGameLib {

    [ServiceContract]
    public interface IMyGameServer {

        [OperationContract]
        void JoinTheGame(string myname, byte[] mypicture,
                          out uint myid, out uint groupid);

        [OperationContract]
        void GetPlayerInfo(uint playerid,
                            out string playername, out byte[] playerpicture);
    }
}

[ServiceBehavior(
    InstanceContextMode = InstanceContextMode.Single)]

public partial class Form1
    : Form, MyGameLib.IMyGameServer
{
public Form1(string subnet) {
    ...
    servicehost = new ServiceHost(this);
    servicehost.AddServiceEndpoint(
        typeof(MyGameLib.IMyGameServer),
        new WSHttpBinding(),
        "http://" + Dns.GetHostName() + ":60000");
    servicehost.Open();
}
}
```
ChannelFactory<IMyGameServer> servicefactory =
    new ChannelFactory<IMyGameServer>(
        new WSHttpBinding(),
        new EndpointAddress(
            "http://" + serveraddress + ":60000"));

IMyGameServer gameserver =
    servicefactory.CreateChannel();

gameserver.JoinTheGame(
    playername, picturebytes, out myid, out groupid);
```
How to use XNA
private Player CreatePlayer(
    uint id, string name, byte[] picture,
    Vector2 position)
{
    Player player = new Player(...,
        new SpriteBatch(graphics.GraphicsDevice),
        Texture2D.FromFile(graphics.GraphicsDevice,
            new MemoryStream(picture)),
        ...);
    ...
}
protected override void Draw(GameTime gameTime)
{
    graphics.GraphicsDevice.Clear(Color.CornflowerBlue);

    foreach (Player player in players.Values)
    {
        player.sprite.Begin(SpriteBlendMode.AlphaBlend);
        player.sprite.Draw(player.texture, player.position, Color.White);
        player.sprite.End();
    }

    base.Draw(gameTime);
}
How to abuse QSM
References
Software (1)

• Visual Studio 2005 Professional Edition: MSDNAA
  

• XNA Framework, XNA Game Studio Express http://msdn2.microsoft.com/en-us/xna/

• Other:
  See the tutorial for details (URL on next page)
Software (2)

• QSM:
  – The QUickSilver Project webpage
  – Download the latest distribution
    (0.12 as of the time of writing this presentation)
  – Report bugs to Krzys
    • krzys@cs.cornell.edu
    • Remember to include logs (see the tutorial for details)
Tutorials (1)

• Video Tutorials on MSDN

• MSDN Documentation (three examples)
  (you can find this content also in your local help documentation installed with XNA)
Tutorials (2)

- QSM Tutorial:  

- QSM API:  
  http://www.cs.cornell.edu/projects/quicksilver/QSM/api/

- QSM User’s Guide:  
Discussion