© Copyright 2003, RiverPoint Group LLC

Advanced Features in Java Applications for Mobile Information Devices

James E. Osbourn

RiverPoint Group LLC

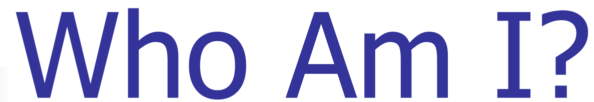
josbourn@riverpoint.com

Who Are You?

- You ...
  - already have attended the first session, or
  - have written MIDP 1.0 applications and want to learn about new MIDP 2.0 features
  - are willing to live on the cutting edge of new platform rollout.
  - Have an idea for a killer wireless Java application.

Colorado Software Summit

© Copyright 2003, RiverPoint Group LLC



Who Am I?

- I …
  - consult IT teams on organization, architecture, methodology, and governance.
  - Have experience as a programmer and architect.
  - Like to keep my hands dirty with the new technologies.
  - Like people, good food, and sharing ideas.

Colorado Software Summit

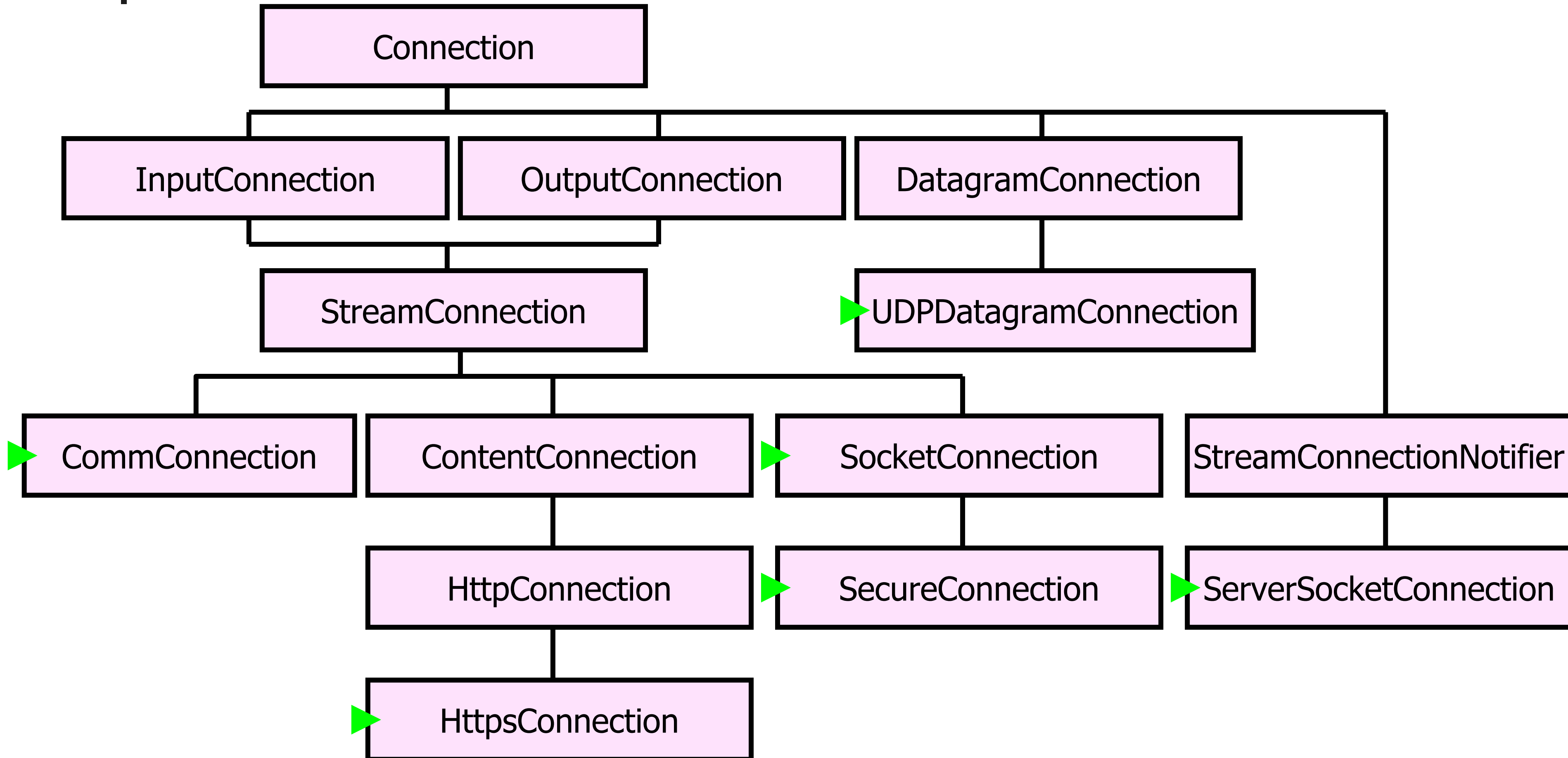
© Copyright 2003, RiverPoint Group LLC

Session’s Contents Include:

- Connectivity
  - HTTP, HTTPS, and more
  - Push
- Canvas
  - Drawing shapes, text & images
  - Blitting, Clipping, & Animating
- Game API
  - Layers and Tiles
  - Sprites & Special Effects
- Sound and Music
  - Players
  - Mobile Media API
- Performance Tuning
  - Diagnosis, Optimizing Memory, Speed Techniques, Buffers
- XML, Ant, Encryption, and Q&A

© Copyright 2003, RiverPoint Group LLC

Connected!



Colorado Software Summit

© Copyright 2003, RiverPoint Group LLC

Connector

- Pass a connection string to Connector’s static method and get back Connection implementation. HTTPS was added to HTTP with MIDP 2.0, but other implementations may be available.
- MIDP HTTP uses a subset of the RFC 2616 standard, only GET, POST, & HEAD commands are required.

© Copyright 2003, RiverPoint Group LLC

It’s Simple

- Pass a URL to Connector’s static open() method. Returned Connection will be an HttpConnection, but you can treat it as an InputConnection. Use the corresponding InputStream and read the data.

```
String url = "http://riverpoint.com/sample";
InputConnection ic = (InputConnection)Connector.open(url);
InputStream in = ic.openInputStream();
// Read stuff from the InputStream
ic.close();
```

Colorado Software Summit

© Copyright 2003, RiverPoint Group LLC

Demo

- Let’s connect our MIDP application and do something. (Gee I hope this works…..)

Colorado Software Summit

© Copyright 2003, RiverPoint Group LLC

POSTing a Form

- Bit more complicated, but here's the deal:
  - Obtain an HttpConnection with open().
  - Modify the header with setRequestMethod() & setRequestProperty().
  - Get an output stream with openOutputStream().
  - Send the request parms on the output stream. See the docs for J2SE java.net.URLEncoder.
  - Read the response with openInputStream().

© Copyright 2003, RiverPoint Group LLC

Catching a Cookie

- Your application may need to handle cookies like a browser would.
  - Check response header with getHeaderField().
  - Save it away for later.

```
InputStream in = hc.opentInputStream();
String cookie = hc.getHeaderfield(“Set-cookie”);
If (cookie != null) {
   int semicolon = cookie.indexOf(‘;’);
   mSession = cookie.substring(0, semicolon);
}
```

  - Send the session ID cookie in the header of subsequent requests with setRequestProperty().

Colorado Software Summit

© Copyright 2003, RiverPoint Group LLC

- Use GET. It's simpler than POST.
- Don't hard code URLs. Duh!
- Use a separate thread for network access. Put up a loading progress indicator.
- Make sure you catch and handle exceptions reasonably. Expect connection failure more often with wireless devices.

© Copyright 2003, RiverPoint Group LLC

HTTPS Is Safer and Just As Easy

- HttpsConnection is an extension of HttpConnection. Adds getPort() to obtain the server's port number (usually 443), and getSecurityInfo().
- SecurityInfo includes certificate information.

```
String url = "https://www.cert.org/";
HttpsConnection hc = (HttpsConnection)Connector.open(url);
SecurityInfo si = hc.getSecurityInfo();
Certificate c = si.getServerCertificate();
String subject = c.getSubject();
```

- Mobile devices can also receive incoming connection requests. A running MIDlet can listen on ServerSocketConnection.
- MIDP 2.0 allows the MIDlet to be launched because of an incoming connection. Called Push.
- Can register at install time with special entries in the .jad file, or at run time with javax.microedition.io.PushRegistry.

© Copyright 2003, RiverPoint Group LLC

Push Registry

- The device AMS is obliged to listen and invoke beyond the life of the MIDlet.

```
PushRegistry.registerConnection(“socket//:80”, MyMIDlet, “*”);
```

- A Web browser pointing at my phone could wake up and invoke a response from my MIDlet. The 3rd parm is a filter, and this one accepts all incoming requests.

© Copyright 2003, RiverPoint Group LLC

Networking Permissions

- The user will be prompted to grant permission. “Always”, “just this one time”, “not this time”, or “shut off wireless Java” are the choices.
- You specify what your MIDlet needs in the descriptor with:
  - MIDlet-Permissions
  - MIDlet-Permissions-Opt

© Copyright 2003, RiverPoint Group LLC

Demo

- Let’s look at another networking sample.

© Copyright 2003, RiverPoint Group LLC

Canvas

- Unlike Screen classes, you must subclass Canvas. But it works nicely with Screens.
- Many of the methods are empty, so you must override them with your own code.
- paint() is abstract and must be defined in your subclass.
- getWidth(), getHeight(), setFullScreenMode()

© Copyright 2003, RiverPoint Group LLC

- Similar to Swing/AWT custom components, you pass a Graphics object to paint.
- To refresh the screen using the current Graphics object use:

```
public void repaint()
public void repaint(int x,int y, int width, int height)
```

- With several repaint() requests, use Display's callSerially().

© Copyright 2003, RiverPoint Group LLC

Graphics

- All drawing takes place in coordinate space based on pixels of the device. Upper-left corner by default. It can be translate().

Colorado Software Summit

Draw and Fill Shapes

- Draw:

```
drawLine(int x1, int y1, int x2, int y2)
drawRect(int x, int y, int width, int height)
drawRoundRect(int x, int y, int width, int height, int arcWidth, int arcHeight)
drawArc(int x, int y, int width, int height, int startAngle, int arcAngle)
```

- Fill:

```
fillTriangle(int x1, int y1, int x2, int y2, int x3, int y3)
fillRect(int x, int y, int width, int height)
fillRoundRect(int x, int y, int width, int height, int arcWidth, int arcHeight)
fillArc(int x, int y, int width, int height, int startAngle, int arcAngle)
```

© Copyright 2003, RiverPoint Group LLC

Demo

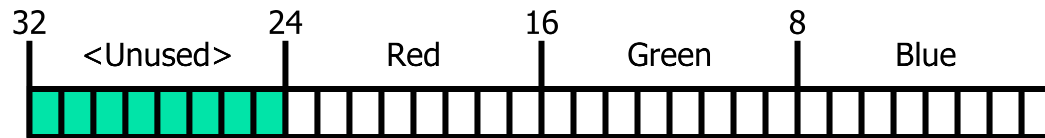
- Let's go draw some shapes.

© Copyright 2003, RiverPoint Group LLC

Color Graphics

- You can set the drawing color with RGB values:

public void setColor(int RGB)



public void setColor(int red, int green, int blue)

- A new method is very handy to find out what a device will do with a color.

public native int getDisplayColor(int color)

© Copyright 2003, RiverPoint Group LLC

Line Style

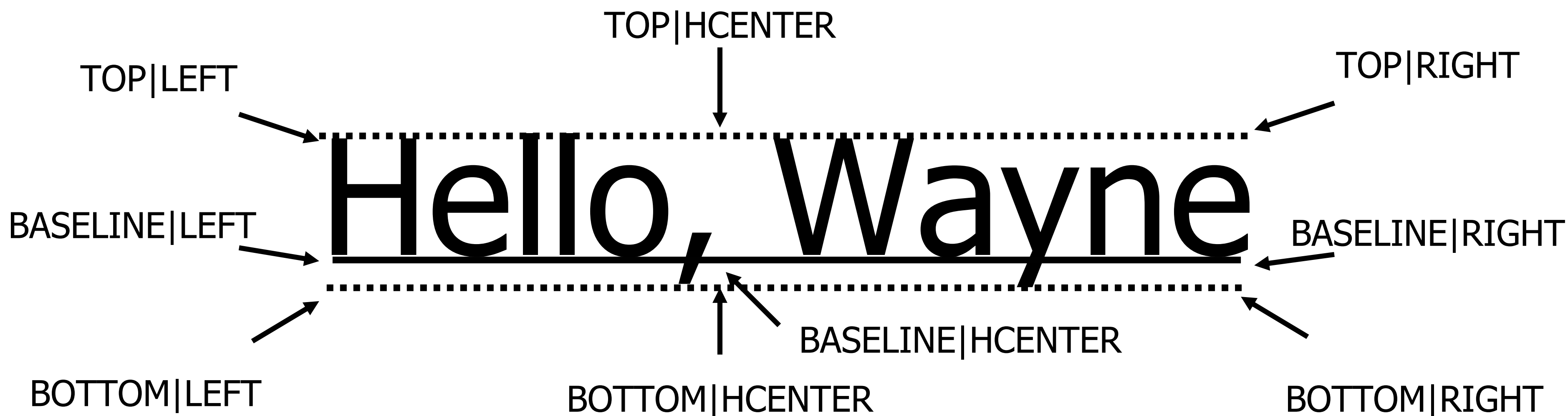
- Called *Stroke Style*, there are two choices: SOLID and DOTTED
- Device decides how the Dotted style will be drawn.

```
getStrokeStyle()
setStrokeStyle()
```

© Copyright 2003, RiverPoint Group LLC

Draw Text

- Draw your text using defined anchor points. Each anchor point is described in terms of horizontal and vertical points.



© Copyright 2003, RiverPoint Group LLC

Draw Text

- Draw characters or Strings using:

```
public void drawChar(char character, int x, int y, int anchor)
public void drawChars(char[] data, int offset, int length,
    int x, int y, int anchor)
public void drawString(String str, int x, int y, int anchor)
public void drawSubString(String str, int offset, int length,
    int x, int y, int anchor)
```

© Copyright 2003, RiverPoint Group LLC

Fonts

- MIDP fonts have *face, style,* & *size*. You may be disappointed by the lack of variety.
- Faces:

  FACE_SYSTEM, FACE_MONOSPACE, FACE_PROPORTIONAL
- Styles:

  STYLE_PLAIN, TYLE_BOLD, STYLE_ITALIC, STYLE_UNDERLINE
- Sizes:

  SIZE_SMALL, SIZE_MEDIUM, SIZE_LARGE

Colorado Software Summit

© Copyright 2003, RiverPoint Group LLC

Fonts

- Use these methods:

  getDefaultFont() provided by the device.

  getFace(), getStyle(), getSize()

  setFace(), setStyle(), setSize()

  isPlain(), isBold(), isItalic(), isUnderlined()

- Create a font by:

```
Font f = Font.getFont(Font.FACE_PROPORTIONAL,
  Font.STYLE_ITALIC, Font.SIZE_SMALL);
```

© Copyright 2003, RiverPoint Group LLC

Fonts

- There are also methods for measuring the text which is useful for laying out a custom canvas.

  getHeight() & getWidth() for lines of text.

  charWidth(), charsWidth(),

  stringWidth() & substringWidth() for parts of the text.

© Copyright 2003, RiverPoint Group LLC

Demo

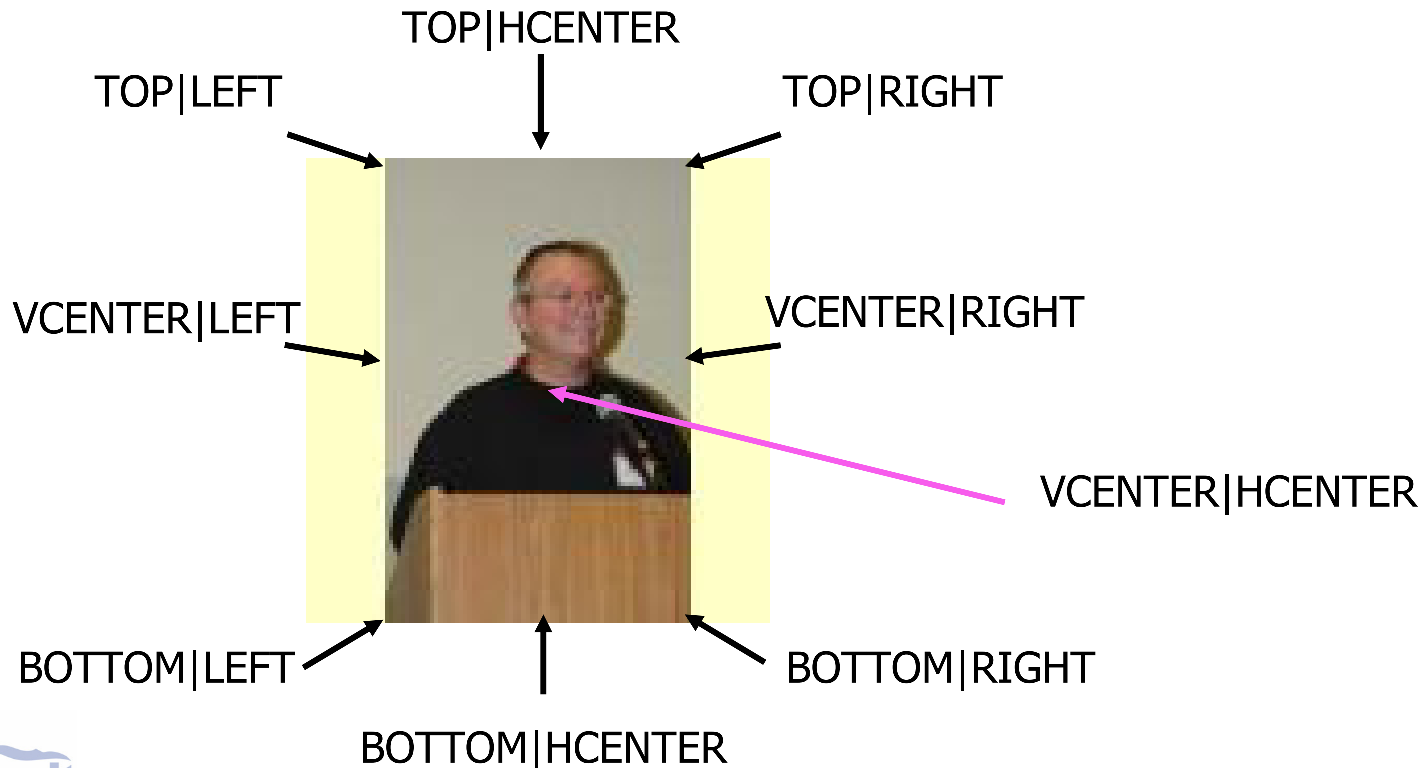
- Draw some text on a canvas, ok?

© Copyright 2003, RiverPoint Group LLC

Draw Images

- Use this one method to draw an image:

  public void drawImage(Image img, int x, int y, int anchor)



© Copyright 2003, RiverPoint Group LLC

Advanced Rendering

- MIDP 2.0 allows rendering a portion of an image and calls for transparency support.

```
public void drawRegion(Image src, int x_src, int y_src,
    int width, int height, int transform, int x_dest, int y_dest,
    int anchor)
```

- Transform parm allows animation frames packed into one image:

```
TRANS_NONE, TRANS_ROT90, TRANS_ROT180,
    TRANS_ROT270, TRANS_MIRROR,
    TRANS_MIRROR_ROT90, TRANS_MIRROR_ROT180,
    TRANS_MIRROR_ROT270
```

© Copyright 2003, RiverPoint Group LLC

Integer Arrays Become Images

- Just as colors can be set as a packed integer, an image can be portrayed as an array of integers, one for each pixel.

```
public void drawRGB(int[] rgbData, int offset,
    int scanlength, int x, int y, int width, int height,
    boolean processAlpha)
```

- The last parm instructs the method to look for each pixel's opacity value in the high-order byte of the integer value.

© Copyright 2003, RiverPoint Group LLC

Blitting

- Copying one region of the screen to another is critical for some games. Use:

```
public void copyArea(int x_src, int y_src, int width, int height, int x_dest, int y_dest, int anchor)
```

- Anchor is the same as drawImage().
- Can not be used on Graphics object that draws directly on the screen. A Graphics on a double-buffered Canvas or a Graphics from GameCanvas's getGraphics() will work.

Colorado Software Summit

© Copyright 2003, RiverPoint Group LLC

Clipping

- Each Graphics has a rectangular clipping shape that limits the drawing area. Use:

  getClipX(), getClipY(), getClipWidth(), getClipHeight()

- Modify the clipping shape with:

  public void setClip(int x, int y, int width, int height);

- Or use another intersecting clip shape to set a new shape:

  public void clipRect(int x, int y, int width, int height);

© Copyright 2003, RiverPoint Group LLC

Keys and Buttons

- Canvas has better support for interacting with individual keys on the device.

  protected void keyPressed(int keyCode)

  protected void keyReleased(int keyCode)

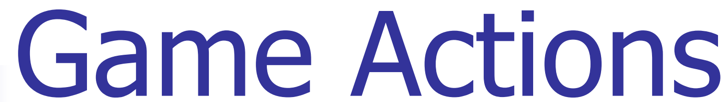
- Each time a key is pressed and released these methods are called with a code.

  KEY_NUM0, KEY_NUM9, KEY_STAR, KEY_POUND, etc.

- Also:

  hasRepeatEvents(), keyRepeated(), getKeyName()

© Copyright 2003, RiverPoint Group LLC



Game Actions

- MIDP tries to save you from being too device specific with getGameAction() which allows you to map a key code and get:

  UP, DOWN, LEFT, RIGHT, FIRE,

  GAME_A, GAME_B, GAME_C, GAME_D
- The implementation will provide this mapping for you.

© Copyright 2003, RiverPoint Group LLC

Pointer Events

- Some devices have pointers, like PDAs. Check for these with:

  hasPointerEvents()

  hasPointerMotionEvents()

- If it does, these methods will get called:

  protected void pointerPressed(int x, int y)

  protected void pointerReleased(int x, int y)

  protected void pointerDragged(int x, int y)

Colorado Software Summit

© Copyright 2003, RiverPoint Group LLC

- Dramatic improvement in quality for the price of a screen's worth of memory.

  isDoubleBuffered()

- If false, then you have to do it yourself.
  - Create an offscreen image with: Image.createImage(int width, int height)
  - Obtain Graphics that draws into the image.
  - Draw stuff into the off-screen image.
  - In Canvas' paint() method, use drawImage().

© Copyright 2003, RiverPoint Group LLC

Demo

- Let’s look at some code for Graphics and Image handling.

© Copyright 2003, RiverPoint Group LLC

Multithreading

- Don’t hold up an event thread! Callbacks should do no more than they must before returning control to the device.
- Let the device do as much for you as it can. Use callSerially() to let the device control the pace and completion of your animation.
- Use a separation thread if you want to control pace from one platform to another.

MIDP 2.0 Game API

- Makes MIDP much stronger platform for playing games.
- Introduces GameCanvas which allows control of exactly when the display is updated and what portion of the screen is updated.
- Other methods deal with Layers which allow composed scenes from different elements.

© Copyright 2003, RiverPoint Group LLC

GameCanvas

- GameCanvas extends Canvas with animation and key state polling. The two are used differently.
  - Subclass Canvas and define a paint() method. Inside paint() use a Graphics to render. Call repaint() and the system calls paint() for you.
  - Subclass GameCanvas. Draw on the screen with the Graphics returned from getGraphics(). Update with flushGraphics() which does not return until the screen is updated.

© Copyright 2003, RiverPoint Group LLC

GameCanvas

- You can make a specific portion update with:

```
public void flushGraphics(int x, int y, int width, int height)
```

- GameCanvas makes it easy to use inside a game loop like:

```
Graphics g = getGraphics();
While(true) {
   // Check for user input
   // Update game state
   // Draw stuff using g
   flushGraphics();
}
```

© Copyright 2003, RiverPoint Group LLC

Demo

- Call the GameCanvas constructor from your subclasses constructor. Pass a boolean to indicate normal key event suppression.
- Let's review a basic sample.

© Copyright 2003, RiverPoint Group LLC

Key State Polling

- Instead of passively waiting for key event callbacks, GameCanvas offers a method that returns the current state of the keys:

  public int getKeyStates()

- Gives your MIDlet more control. Returned integer uses one bit to represent each of the game actions. This allows you to poll in the game loop instead of relying on a callback in a separate thread.

© Copyright 2003, RiverPoint Group LLC

Key State Polling

GameCanvas constants:

- UP_PRESSED
- DOWN_PRESSED
- LEFT_PRESSED
- RIGHT_PRESSED
- FIRE_PRESSED
- GAME_A_PRESSED
- GAME_B_PRESSED
- GAME_C_PRESSED
- GAME_B_PRESSED

```
Graphics g = getGraphics();
While(true) {
// Check for user input.
int ks = getKeyStates();
if ((ks & UP_PRESSED) != 0)
  moveUp();
else if ((ks & Down_PRESSED) != 0 )
  moveDown();
// …
// Update game state.
// Draw stuff using g.
flushGraphics();
}
```

© Copyright 2003, RiverPoint Group LLC

Layer

- Like hand-drawn animation, Layer lets you create the screen with background scenes and smaller stuff in the foreground on transparent overlays
- Layer is abstract with 2 concrete subclasses. It has location, size, and visibility.

```
public final int getX()          public final int getY()
public final int getWidth()      public final int getHeight()
public void setPosition(int x, int y)
```

© Copyright 2003, RiverPoint Group LLC

Layer

- Layer also gives a handy method for a relative move.

```
public void move(int dx, int dy)

public final boolean isVisible();
public void setVisibile(boolean visible);
public abstract void paint(Graphics g);
```

© Copyright 2003, RiverPoint Group LLC

Managing Layers

- LayerManager keeps an ordered list of layers. The index indicates their front-to-back order. Use:

```
public void append(Layer l)
public void remove(Layer l)
public void insert(Layer l, int index)
public void getLayerAt(int index)
public int getSize()
```

- LayerManager includes the idea of a view window on a scene that is larger than the screen. Default origin is 0, 0 and view window is as large as possible.

```
public void setViewWindow(int x, int y,
    int width, int height)
```

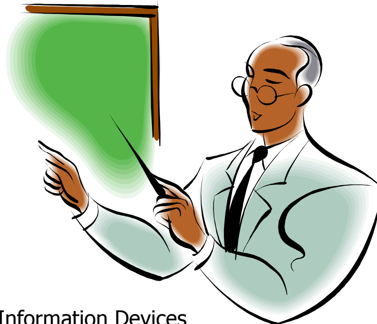
- Call paint to draw the scene:

```
public void paint(Graphics g, int x, int y)
```

© Copyright 2003, RiverPoint Group LLC

Tiled Layers

- Single Image divided into equal-sized pieces. Drawn on a Graphics object with paint() inherited from Layer.
- The divided Image is numbered (indexed) starting with 1, row and column numbers start with 0.
- Confusing? Look at this example of a tiled Image.



© Copyright 2003, RiverPoint Group LLC

Tiled Layers

- The size of the tiled layer is:

  Width = [number of column] x [tile width]

  Height = [number of rows] x [tile height]

- Construct a TiledLayer with:

```
public TiledLayer(int columns, int rows,
    Image image, int tileWidth, int tileHeight)
```

- Change these sets of tiles with:

```
Public void setStaticTileSet(Image image,
    int tileWidth, int tileHeight)
```

Colorado Software Summit

© Copyright 2003, RiverPoint Group LLC

Tiled Layers

- Other methods:

```
public final int getColumns()
public final int getRows()
public final int getCellWidth
public final int getCellHeight()
public void setCell(int col, int row, int tileIndex)
public void fillCells(int col, int row, int numCols,
    int numRows, int tileIndex)
```

© Copyright 2003, RiverPoint Group LLC

Animated Tiles

- A virtual tile whose mapping is changed at runtime. Instead of calling setCell() on all the cells you want to change, use:

  ```
  public int createAnimatedTile(int staticTileIndex)
  ```

- Pass the first tile of the animation & use the return to setCell(), then change contents:

  ```
  public void setAnimatedTile(int animatedTileIndex,
      int staticTileIndex)
  ```

© Copyright 2003, RiverPoint Group LLC

Demo

- Let's run a sample and build a tiled Layer.

© Copyright 2003, RiverPoint Group LLC

Sprites

- Used to animate a layer that is the same size as a single tile, called a frame. Constructed from a source image that is divided in equal frames. You must pack the frames into Image.

```
public Sprite(Image image, int frameWidth, int frameHeight)
```

- If you need to change the image:

```
public void setImage(Image image, int frameWidth,
   int frameHeight)
```

© Copyright 2003, RiverPoint Group LLC

Animating Sprites

- Sequence frames. Default is all the frames in the source Image. Index starts at 0.

  public void setFrameSequence(int[] sequence)

- You must tell the frames to change. Use a separate animation thread.

  nextFrame(), previousFrame(), getFrame(), getFrameSequenceLength()

  - There is no getFrameSequence(), so save it when you set it.

 © Copyright 2003, RiverPoint Group LLC

Transforming Sprites

- Saves much effort by allowing sprites to turn around without creating all the directional images.

TRANS_NONE
TRANS_ROT90
TRANS_ROT180
TRANS_ROT270

TRANS_MIRROR
TRANS_MIRROR_ROT90
TRANS_MIRROR_ROT180
TRANS_MIRROR_ROT270

© Copyright 2003, RiverPoint Group LLC

- Tricky during transformations because the reference pixel doesn’t move. Default 0, 0.
- To adjust the reference point:

  public void defineReferencePixel(int x, int y)

- Use these to find the reference point:

  getRefPixelX() getRefPixelY()

- Or position the Sprite’s reference point:

  Public void setRefPointPosition(int x, int y)

© Copyright 2003, RiverPoint Group LLC

Collisions

- Did the Sprite hit that wall? Two ways to tell:
  - Compare rectangles representing the sprite and the wall. If they intersect, we have collision. May not be realistic for non-rectangular shapes.
  - Compare each pixel of the sprite and the wall. If two opaque pixels overlap, we have collision. Very realistic, but involves more computation.
- Change the collision rectangle with:

```
public void defineCollisionRectangle(int x, int y,
        int width, int height)
```

Colorado Software Summit

© Copyright 2003, RiverPoint Group LLC

Collisions

- Sprite can detect collisions with other sprites, TiledLayers, and Images:

public final boolean collidesWith(Sprite s, boolean pixelLevel)

public final boolean collidesWith(TiledLayer t, boolean pixelLevel)

public final boolean collidesWith(Image image, int x, int y, boolean pixelLevel)

  - Sprite: collision rectangles, or pixels inside the collision rectangles
  - TiledLayer: collision rectangle of sprite to the tiles, or pixels inside the collision rectangle to the tiles
  - Image: collision rectangle of sprite to the Image bounds, or pixels inside the collision rectangle to the pixels in the Image

© Copyright 2003, RiverPoint Group LLC

Spawning Sprites!

- Sprite has a copy constructor:

  ```
  public Sprite(Sprite s)
  ```

- Wow, creates a new Sprite with all the attitude of the original:
  - ✓Source image frames
  - ✓Frame sequence
  - ✓Current frame
  - ✓Current transformation
  - ✓Reference pixel
  - ✓Collision rectangle

Colorado Software Summit

© Copyright 2003, RiverPoint Group LLC

Special Effects

- Display has a couple of methods that are fun for gamers.

```
public boolean flashBacklight(int duration)
public boolean vibrate(int duration)
```

Colorado Software Summit

© Copyright 2003, RiverPoint Group LLC

Demo

- Let’s play a game…..

© Copyright 2003, RiverPoint Group LLC

- MIDP 2.0 media APIs are a subset of Mobile Media API.
- Subset is the Audio Building Block and includes playing simple tones and sampled audio.

```
public static void playTone(int note, int duration,
    int volume)
```

© Copyright 2003, RiverPoint Group LLC

Get a Player

- Just ask Manager for one:

```
URL url = "http:/riverpoint.com/res/relax.wav";
Player p = Manager.createPlayer(url);
p.start();
```

- Or if you don't have a server to provide the content type:

```
InputStream in =
   getClass().getResourceAsStream("/relax.wav");
Player p = Manager.createPlayer(in, "audio/x-wav");
p.start();
```

© Copyright 2003, RiverPoint Group LLC

Audio Content Types

- Really just a file specification to tell how the data makes the sound. Specified by MIME types. Player knows how to render audio data. Manager hands out the right Player for the content type. 8-bit PCM WAV must be supported if the device does sampled audio at all. Ask the Manager:

```
public static String getSupportedContentTypes(String protocol)
public static String getSupportedProtocols(String contentType)
```

© Copyright 2003, RiverPoint Group LLC

Player

- Player states progress as follows:
  - UNREALIZED, created but hasn't obtained the data
  - REALIZED, has found the data source
  - PREFETCHED, has fully prepared to render
  - STARTED, has begun to render the audio
  - CLOSED, has released all resources and can not be used

```
public void realize()
public void prefetch()
public void start()
public void close()
public void deallocate()
public void stop()
public int getState()
```

© Copyright 2003, RiverPoint Group LLC

Player Events

- During the Player's life, it will post several events, so there are listeners:

```
public void addPlayerListener(PlayerListener playerlistener)
public void removePlayerListener(PlayerListener playerlistener)
```

- PlayerListener has 1 method:

```
public void playerUpdate(Player player, String event,
    Object eventData)
```

- Common Events:

STARTED, END_OF_MEDIA, VOLUME_CHANGED, etc.

© Copyright 2003, RiverPoint Group LLC

Demo

- Let’s play some tunes….

© Copyright 2003, RiverPoint Group LLC

Performance Tuning

- Processor is likely to be slower than your development PC. Skinny and fast is important.
- However, write your code for clarity and maintainability first, then optimize if it needs it. Having a target platform and unbiased user is valuable.
- Benchmark, then improve.

© Copyright 2003, RiverPoint Group LLC

- Most of the J2SE tools are not available for J2ME, so do it by hand:

```
public long freeMemory()
public long totalMemory()
```

- Total memory can change if the device gives the MIDP application more.
- Use the clock call in java.lang.System:

```
public static long currentTimeMillis()
```

© Copyright 2003, RiverPoint Group LLC

Benchmarking

- Memory use check:

```
Runtime runtime = Runtime.getRuntime();
long before, after;
System.gc();
before = runtime.freeMemory();
Object newObject = new String();
after = runtime.freeMemory();
long size = before – after;
```

© Copyright 2003, RiverPoint Group LLC

Benchmarking

- Calculate execution time:

```
long start, finish;
start = System.currentTimeMillis();
someMethod();
finish = System.currentTimeMillis();
long duration = finish – start;
```

© Copyright 2003, RiverPoint Group LLC

Diagnostic Tools in J2MEWTK

- Let's look at what the toolkit can do….

© Copyright 2003, RiverPoint Group LLC

Respect Memory Use

- Easy to rely on the garbage collector in J2SE, but object allocation and garbage collection can really hurt your small device performance.
- Don't allocate new objects inside of a loop.
- Don't concatenate strings inside of a loop.
- Be prepared for memory failure. For every object creation, catch:

java.lang.OutOfMemoryError

© Copyright 2003, RiverPoint Group LLC

Respect Memory Use

- Try to avoid Hashtable or Vector and use an array if you can. If you must use Hashtable or Vector, try to size them correctly when you create them.
- Work directly with a Record’s byte array and avoid wrapping it.
- Read and write whole arrays, not one byte at a time. Be prepared to buffer your I/O.

© Copyright 2003, RiverPoint Group LLC

Clean Up Resources

- Release resources as soon as you are done with them. Set internal arrays to null. Even call System.gc() if you need to.
- Drop network connections when you're done with them. Use the finally block to assure it is not left as a hanging connection after some exception.
- This idea applies to all streams, RecordStores, record enumerations, *etc*.

Colorado Software Summit

© Copyright 2003, RiverPoint Group LLC

Keep the User Happy

- Even if you have to take time to perform some function, optimize the perceived speed by keeping the screen busy. Make sure they know it's still alive.
- Users think of their mobile device as an appliance that is supposed to work right all the time, not like their PC.

© Copyright 2003, RiverPoint Group LLC

Optimize Around Your MIDlet

- Make the MIDlet Suite (.jar) small with a good obfuscator.
- Include only classes that you need.
- Partition seldom used functions so they don’t get loaded unless they are asked for.

© Copyright 2003, RiverPoint Group LLC

XML, Ant, Encryption *et al.*

- To finish I'd like to open a discussion of other topics here at the conference this week and how they might apply to MID programming.

Colorado Software Summit

© Copyright 2003, RiverPoint Group LLC

- Please fill out your evaluation form now.
- I am a first-year presenter and really need your feedback.
- If I didn’t get to something you wanted to cover, please ask me at the fireplace.