

Domain-Driven Design, Eric Evans, 2004, Addison-Wesley

Patterns of Enterprise Application Architecture, Martin Fowler, 2003, Pearson Education

Android Video
Main Classes

MediaPlayer
   Plays video & audio
   Bit complex to use directly

VideoView
   Wraps Media player in a view
   Simpler to use, but less functionality

MediaController
   Displays controls to start, stop, move location in video

For an example of using Media Player directly see https://github.com/commonsware/Android/tree/master/vidtry/Player.java
MediaPlayer & Streaming

Player has to buffer video/audio before playing

Can't start video until ready

Use PreparedListener to find out when video is ready
Example Using VideoView

VideoView id "video"

EditText id "questionField"

MediaController
Not in layout
public class MainActivity extends Activity implements MediaPlayer.OnPreparedListener{
    MediaPlayer mediaPlayer;
    private VideoView video;
    private EditText questionField;
Setting up the VideoView

@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    questionField = (EditText) findViewById(R.id.question);

    String url = "http://www-rohan.sdsu.edu/~whitney/audio/courses/fall12/cs580/
    cs580_10_11_12.mp4";

    video = (VideoView) findViewById(R.id.video);
    video.setOnPreparedListener(this);
    video.setVideoPath(url);
    MediaController controls=new MediaController(this);
    video.setMediaController(controls);
    video.requestFocus();
}

Starting the Video

public void onPrepared(MediaPlayer player) {
    player.seekTo(1000); // just to show how start at different location.
    player.start();
}

How to Display Questions at Correct time

Periodically
   Look at current position of video
   See if any questions are near current position

Don't forget to turn off periodic checks when not needed.

Periodic check with
   Timer
       More involved

   postDelayed
       Requires more time on UI thread
Question Class

Each question object has:

    text
        String - text of question

    time
        Location in video to show question
private class DisplayQuestions implements Runnable {

    public int lookAheadDuration() {
        return 1000*30;
    }

    @Override
    public void run() {
        int now = video.getCurrentPosition();
        questionField.setText(""),
        int windowEnd = now + lookAheadDuration();
        for (Question x : questions) {
            if ((now < x.time()) && (x.time() < windowEnd))
                questionField.setText(x.text());
        }
        questionField.postDelayed(new DisplayQuestions(), lookAheadDuration());
    }
}
Start it off

public void onPrepared(MediaPlayer player) {
    player.seekTo(1000); //just to show how start at different location.
    player.start();
    questionField.post(new DisplayQuestions());
}
Issues to handle

When device is rotated video restarts from beginning

If activity is killed in background video restarts from beginning
GUI Design
Recommended Reading – Designing GUIs

User Interface Design for Programmers, Joel Spolsky, 2001

There is a printed longer version of the book. The on-line version is free and will get you started.

Design of Everyday Things, Donald Norman, 1990

This is an excellent book, is entertaining and only costs $12 new at Amazon. Anyone that designs or builds anything has to read this book.

These books do not cover the mechanics building a GUI. They do not cover which fonts and colors to use. They just get you started thinking about the really important questions related to GUI design.
Psychopathology of Things

Affordance
Conceptual Model
Make Things Visible
Mapping
Feedback
Keys

[Image of two keys]
Affordance
Affordance
Make Things Visible

"Push-button keyless start couldn't be simpler"
Improved Version

Photo http://www.edmunds.com/ownership/audio/articles/106651/article.html
Mapping
Mapping
Software Structure with UI & Database

Tiers – parts of program run on different machines

Layers – parts of program run on same machine

Layers

Presentation – Display of Data

Domain – Logic related to purpose of the application

Data Source – Communication with data source
Keep presentation & domain layers separate

GUI code should not contain domain logic

In simple cases different methods in one class may handle presentation and domain logic

A single method does either presentation or domain logic

Can you test the domain logic with automated unit testing
public class TimeDateClient{
    private static final char CARRIAGE_RETURN = (char) 13;
    private static final char LINE_FEED = (char) 10;
    String server;
    int serverPort;

    public TimeDateClient(String serverNameOrIP, int port){
        server = serverNameOrIP;
        serverPort = port;
    }

    public String date() throws IOException {  return sendMessage("date");}

    public String time() throws IOException{ return sendMessage("time"); }

    public String sendMessage(String message) throws IOException{
        Socket serverConnection = new Socket(server, serverPort);
        writeMessage(message, serverConnection);
        byte[] result = readMessage(serverConnection);
        serverConnection.close();
        return new String(result);
    }
}
private byte[] readMessage(Socket serverConnection) throws IOException {
    UpToFilterInputStream in = new UpToFilterInputStream(
            new BufferedInputStream(
                    serverConnection.getInputStream()));
    byte[] result = in.upTo(LINE_FEED);
    return result;
}

private void writeMessage(String message, Socket serverConnection) throws IOException {
    OutputStream out = new BufferedOutputStream(
            serverConnection.getOutputStream());
    out.write((message + CARRIAGE_RETURN).getBytes());
    out.flush();
}
Now add a GUI that uses Domain layer
Example: Time Date Client

Protocol has two messages
Date
Time

Client has one button to get time & date
Smart UI Pattern

“the separation of UI and domain is so often attempted and so seldom accomplished that its negation deserves a discussion”
   Eric Evans, Domain-Driven Design

The Pattern
Put all business logic into user interface

Divide the application into different small user interfaces

Use relational databases as back end

Use automated UI building and visual programming tools
Smart UI Pattern

Advantages

- High and immediate productivity for simple applications
- Little training need by the developer
- Short development time for small modules

Disadvantages

- No reuse – code gets duplicated
- Integration of applications difficult
- Very difficult to add new functionality to existing application
- Difficult to build complex applications
GUI Clients & Servers

GUI Clients
  Used to get work done
  Good when they help people get work done

Users do not care about the client-server protocol

Don’t expose the user to the client-server protocol

Build the GUI to help people accomplish a task, not just to perform the client-server protocol
Interface Design When You Don't Know How
Basic Rule for Good Visual Design

Hire a graphic/GUI designer
Basic Rule for Almost Everything Else

Painstakingly follow established standards
What Makes a Good GUI?

A user interface is well-designed when the program behaves exactly how the user thought it would.

Deleting a file on a Mac

Move it to the trash can!

How do you unmount floppies & external hard drives?

Move it to the trash can

But users think this will delete or erase it
Mental Models & Users

Users don’t read the manual

More text on the screen - fewer people will read it

Users have a mental model of how your program works

Make sure your program fits their mental model

Users think the trash can deletes things
The Process of Creating a GUI

Plan ahead
Use bite-sized chunks
Abandon the waterfall life cycle in favor of iterative design
Conduct user testing early and often
Focus on the users' needs and involve them in the process
Come up with good, testable usability goals
Hire a graphic designer
Why iterative design?
But we don't have time to redo the GUI!
Prototype the GUI?
Problems with Software Prototypes

- Software Prototypes take too long to build and change
- Testers tend to comment on "fit and finish" issues
- Developers resist changes
- Managers ship the prototype
- Software Prototypes set false expectations
- Single bug in a software prototype can halt a test
Use Paper Prototypes
Paper Prototype Kit

White paper
  Unlined
  Heavy enough to endure repeated testing and revisions

5-by-3-inch cards
  Use as construction material
  Taking notes

Adhesives
  Tape: clear, colored, double-backed, etc.
  Glue sticks
  Post-It glue
  White correction paper

Markers
  Colored pens and pencils
  Highlighters
  Fine and thick markers

Sticky note pads
Acetate sheets
Scissors,
X-acto knives,
straightedges,
Band-Aids
White-out

Tuesday, October 16, 12
Build A Paper Prototype

Do it fast

Construct Models, not illustrations
Test your Prototype with Users
Preparing for a Test

Select your users

Know the demographics of your users

Testers should represent the whole range of users

Prepare test scenarios

Practice
Conducting a Test

Greeter
  Welcomes users
  Puts users at ease

Facilitator
  Runs the test
  Gives users instruction
  Encourage the user to express their thoughts

Computer
  Runs the computer

Observer
  Takes notes

Tuesday, October 16, 12
The Test

Video tape the test

Before starting the test, explain the process to the user

Debrief the user after the test

Evaluate the Results