CS342: Software Design

September 4, 2017

Agenda

Web apps vs. “device” apps

Java coding by example

- Class and interface
- Event driven programming
- Introduction to Java graphics
- Multi-thread

Discussion of a product feature and approach to tackle it

Software topics

How to lead a software development project (not project management)

Cloud computing

Software development team and organization

Mobile web app vs native

Technology solution stack (LAMP, JAM, WINS, MEAN)

Build Enterprise software

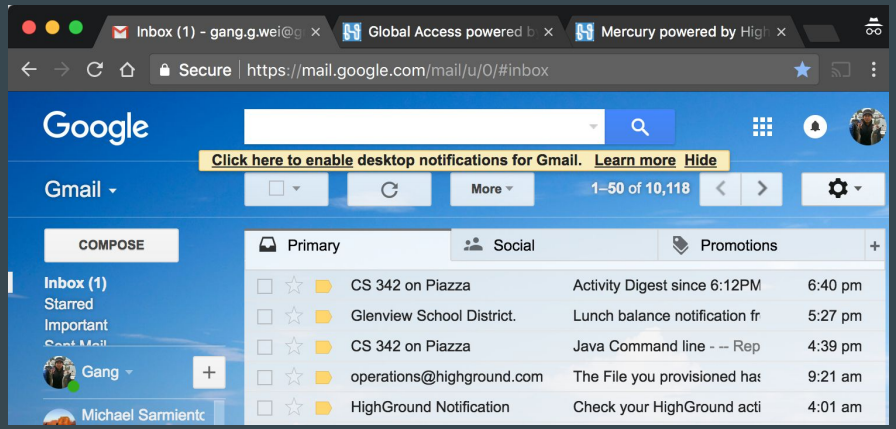
Release cycle

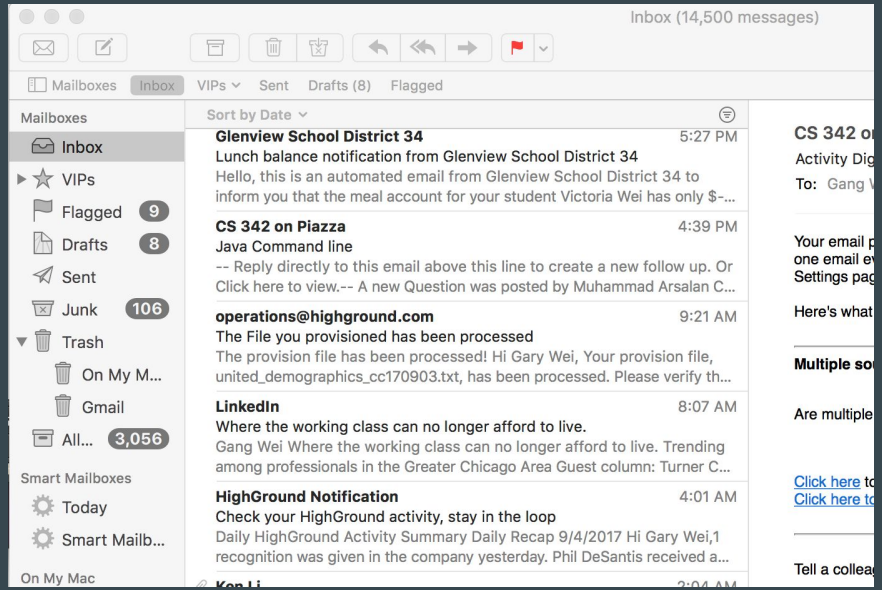
Service oriented architecture and Micro service architecture

Web application vs. “Device” application

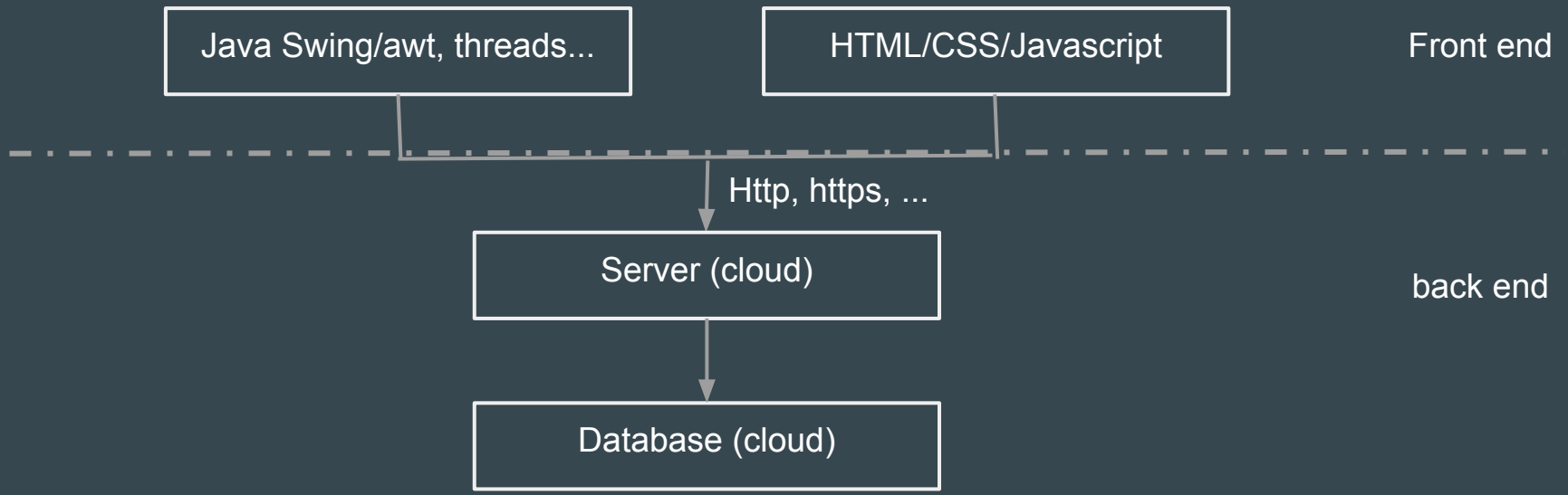
Accessed through web browsers

Downloaded and installed on your computer or mobile device





Web and device alike, most applications require internet access nowadays



Which one is better?

Performance?

Better control?

Security

More sticky

System upgrade

Who’s winning?

HelloJava4.java

```java
//file: HelloJava4.java
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class HelloJava4
{
    public static void main( String[] args ) {
        JFrame frame = new JFrame( "HelloJava4" );
        frame.add( new HelloComponent4("Hello, Java!") );
        frame.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        frame.setSize( 300, 300 );
        frame.setVisible( true );
    }
}
class HelloComponent4 extends JComponent
    implements MouseMotionListener, ActionListener, Runnable
{
    String theMessage;
    int messageX = 125, messageY = 95; // Coordinates of the message
    JButton theButton;
    int colorIndex; // Current index into someColors.
    static Color[] someColors = {
    Color.black, Color.red, Color.green, Color.blue, Color.magenta };
    boolean blinkState;
    public HelloComponent4( String message ) {
        theMessage = message;
        theButton = new JButton("Change Color");
        setLayout( new FlowLayout( ) );
        add( theButton );
        theButton.addActionListener( this );
        addMouseMotionListener( this );
        Thread t = new Thread( this );
        t.start( );
    }
    public void paintComponent( Graphics g ) {
        g.setColor(blinkState ? getBackground( ) : currentColor( ));
        g.drawString(theMessage, messageX, messageY);
    }
    public void mouseDragged(MouseEvent e) {
        messageX = e.getX( );
        messageY = e.getY( );
        repaint( );
    }
    public void mouseMoved(MouseEvent e) { }
    public void actionPerformed( ActionEvent e ) {
        if ( e.getSource( ) == theButton )
            changeColor( );
    }
    synchronized private void changeColor( ) {
        if (++colorIndex == someColors.length)
            colorIndex = 0;
        setForeground( currentColor( ) );
        repaint( );
    }
    synchronized private Color currentColor( ) {
        return someColors[colorIndex];
    }
    public void run( ) {
        try {
        while(true) {
            blinkState = !blinkState; // Toggle blinkState.
            repaint( ); // Show the change.
            Thread.sleep(300);
        }
        } catch (InterruptedException ie) { }
    }
}
```

Highlights of code example

Use IDE vs. plain editor and command line.

Public class should be declared in a file with the same name

Main function. Public, static, void.

You can define HelloComponent4 in a separate file. But need to import proper packages

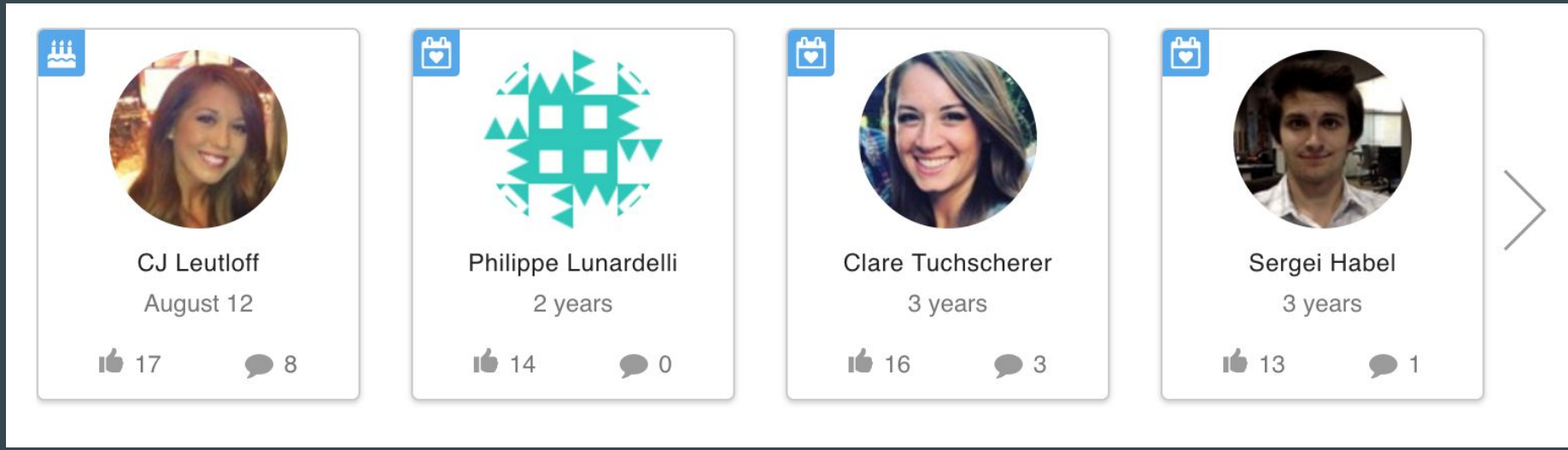
Why in different file?

HelloComponent4 constructor

Highlights of code example (continued)

- JFrame, JComponent
- Extends vs. implements
- Button: addActionListener. Mouse: addMouseMotionListener
- instantiate a thread, and start thread. Thread: share the same global variable and other resources. What's going to happen without thread in this example?
- run()
- Synchronized method
- paintComponent
- mouseDragged

A birthday/anniversary announcement feature



Requirements

- Display Birthday/anniversaries of UIC students/professors of past 7 days
- People are sorted in alphabetical order by last name
- Display 4 persons at a time.
- Announcement type, picture, # of years, etc
- Don’t display “0 year” anniversary
- If the student doesn’t want to show real image, app generates a fake picture
- “Unsubscribe” button
- You can’t unsubscribe your professor who has taught you
- “Next” button. Disabled when this is the last student
- “Previous” button. Disabled when this is the first student

UI: Swing and AWT, etc

- Display frame or window
- Buttons (conditionally disabled)
- Picture display
- Labels: data will come from objects. Need to be formatted
- Icons
- etc.

```
public static void main( String[] args ) {
    JFrame frame = new JFrame( "Announcement" );
    frame.add( new AnnouncementComponent("Birthday and Anniversary") );
    frame.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
    frame.setSize( 300, 300 );
    frame.setVisible( true );
```

Application logic

- Class: People.
- Data properties: Name, picture path, user setting, birthday, anniversary...
- Static method: get next 4 announcements (in date range)
- Method: unsubscribe
    - Validation: check if the person i'm trying to unsubscribe is my teacher
- Method: suppress my picture
- Method: generate random picture

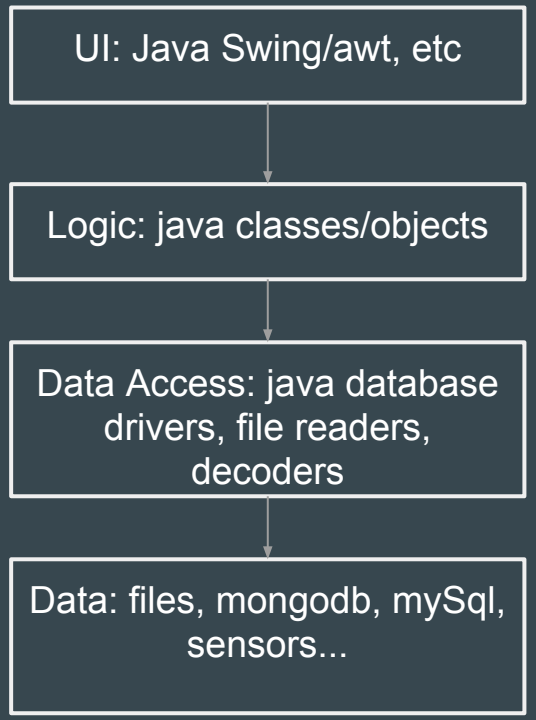
Data access

- Text/csv file: all people info is stored in files. Need to implement file reader
- Relational database: create db connection, run query, write records
- Mongodb:
- Stored in memory or cache

```
DataRange dateRange = new DateRange(today(), today() - 7);
UserDal userDal = new UserDal();
User [] users = userDal.GetBirtdayAnniversaryUsers();
```

Separation of concerns (vertically)

- UI (front end) should be dummy. No logic. Faithfully render data and invoke actions
- Logic: no awareness of UI or database. Only takes care of application logic
- Data access: encapsulate details about data writing/reading
- Data: text file, in-memory objects, SQL db, non-SQL db, images/video



Vertically... is there a “horizontally”?