Writing Brush Handwritten Evaluation APP

—— Graduation Project

Student: Gu Jiapan 1120141268

Supervisor: Prof. Xie Xiang, Zhou Zhiying (abroad)

The graduation project was carried out
under the 3+1+1 Educational Framework at
National University of Singapore (Suzhou) Research Institute

Background

Information age: digital development
Calligraphy : a vital part of Chinese culture

Their combination is imperative

Traditional calligraphy practice

Conditions

- Exquisite tools
- Special environment

Assessment

- Cost of money and time
- Unified standard

Environment

- Cost of money
- Environmentally friendly

Objective

Academic Year	AY 2017/2018 (September 2017 – May 2018)
Title of project	Writing brush handwritten evaluation APP
Abstract (100 – 300 words)	This project aims to design a APP for writing brush handwritten evaluation. This project tasks include: 1) Writing part: When writing brush is used on capacitive screen pad, the coordinate and the size of contact area can be obtained from the API. Utilize the size cue to make handwriting approximate reality. 2) Evaluation part: Compare the handwriting with templates, giving a matching score.
Supervisor	Prof Zhi Y. Zhou
Laboratory facilities to be used	Interactive Media Research Centre, NUSRI
Number of students for the project	1
CA1 requirement	Finish writing part. Utilize the size cue to make handwriting approximate reality.
CA2 requirement	Finish evaluation part. Give a reasonable matching score.

Execution Process

1) Preparatory work:

- ✓ Existing APP on the market
- ✓ Relevant literature

2) Writing part:

- ✓ A writing board demo
- ✓ Android framework and resources
- ✓ Environment configuration
- ✓ Java language
- ✓ Relevant library functions

3) Evaluation part:

- ✓ Graphic-based preparation
- ✓ Algorithm implementation
- ✓ Using effect test

4) Finishing touches:

- ✓ User interface
- ✓ Other improvements

Demonstration

A Writing Part

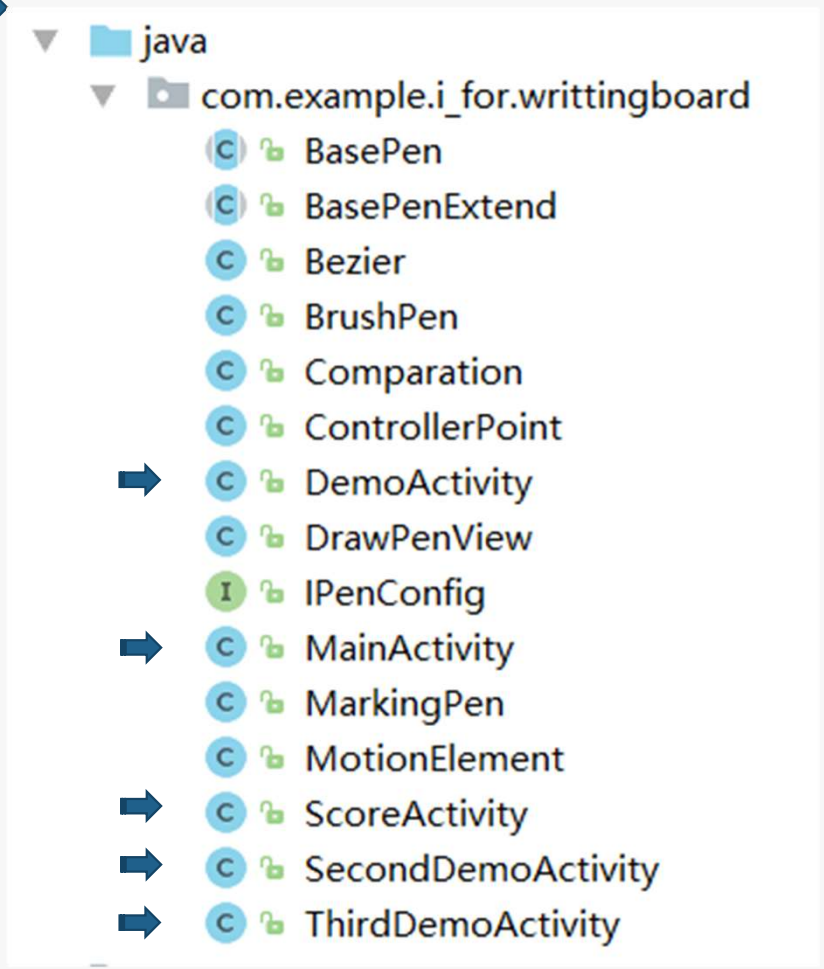
- Activity Description
- User interface Design
- Gesture Recognition and Calligraphy Simulation

Activity Description —— Overview

“MainActivity”

1) “DemoActivity” —— “食”
2) “SecondDemoActivity” —— “家”
3) “ThirdDemoActivity” —— “复”

“ScoreActivity”



Activity Description —— Activity conversion

```java
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;

public class MainActivity extends AppCompatActivity implements View.OnClickListener {
    private Button mBtn;
    @Override
    protected void onCreate (Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        findViews();
        doSomeThing();
    }
    private void doSomeThing() {
        mBtn.setOnClickListener(this);
    }
    private void findViews() {
        mBtn = findViewById(R.id.btn);
    }
    @Override
    public void onClick(View view) {
        Intent intent = new Intent( packageContext: MainActivity.this,DemoActivity.class);
        startActivity(intent);
    }
}
```

User Interface Design

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    ...>
    <LinearLayout
        ...>
        <TextView
            ...>
        <LinearLayout
            ...>
            <Button
                .../>
        </LinearLayout>
        ...
    </LinearLayout>
</android.support.constraint.ConstraintLayout>
```


```
<Button
android:id="@+id/btn_shi"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="食"
android:textSize="30sp"/>
```

Gesture Recognition and Calligraphy Simulation

“DemoActivity” *new* “DrawPenView”

extend View

import Graphics

(Android Library)

invoke “IPenConfig”

(Interface)

new “BrushPen”

“MarkingPen”

extend *(Abstract)* “BasePenExtend”

extend “BasePen”

(Abstract)

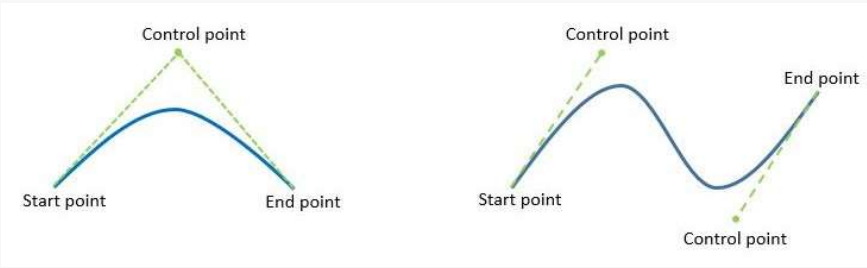
int PEN_WIDTH = 80;
float STROKE_FACTOR = 0.02f;
float VELOCITY_FACTOR = 40f;
int STEP_FACTOR = 10;

invoke “MotionElement”

“Up”, “Down”, “Move”, etc.

new “Bezier”

“ControllerPoint”



B

Evaluation Part

- Graphic-based Preparation
- Algorithm Based on Filling rate
- Algorithm for Skeletonization
- Extracting Strokes
- Algorithm Summary

Graphic-based Preparation

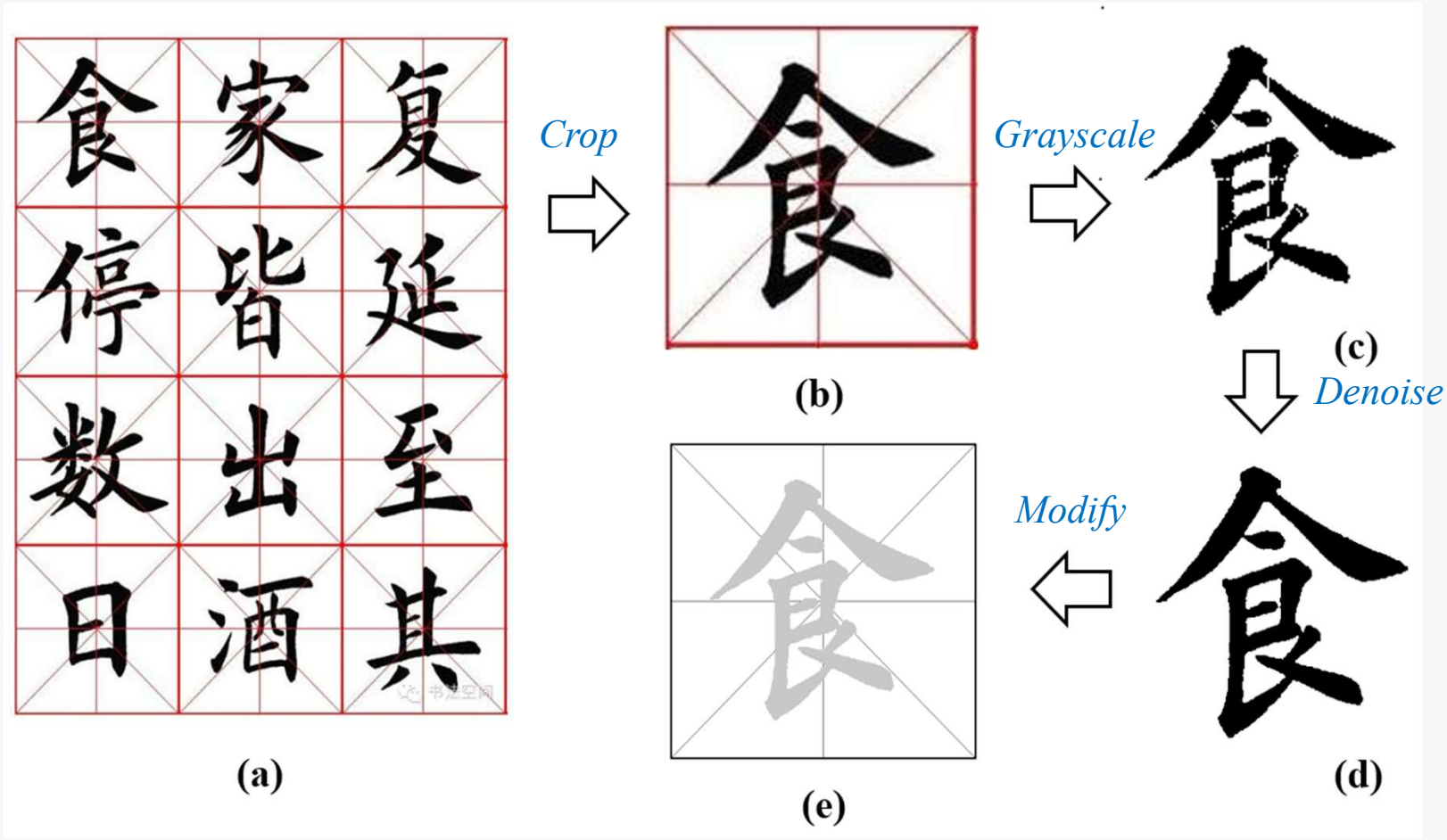
Basic operation based on MATLAB:

1. **Read a picture**:
   "imread ('path')"
2. **Open and show a picture**:
   "imshow (variable)"
3. **Zoom in or out of a picture**:
   "imresize (variable, magnifacation)"
4. **Crop a picture**:
   "imcrop (variable, coordinates of the starting and ending points)"
5. **Save a picture**:
   "imwrite (variable, 'path')"

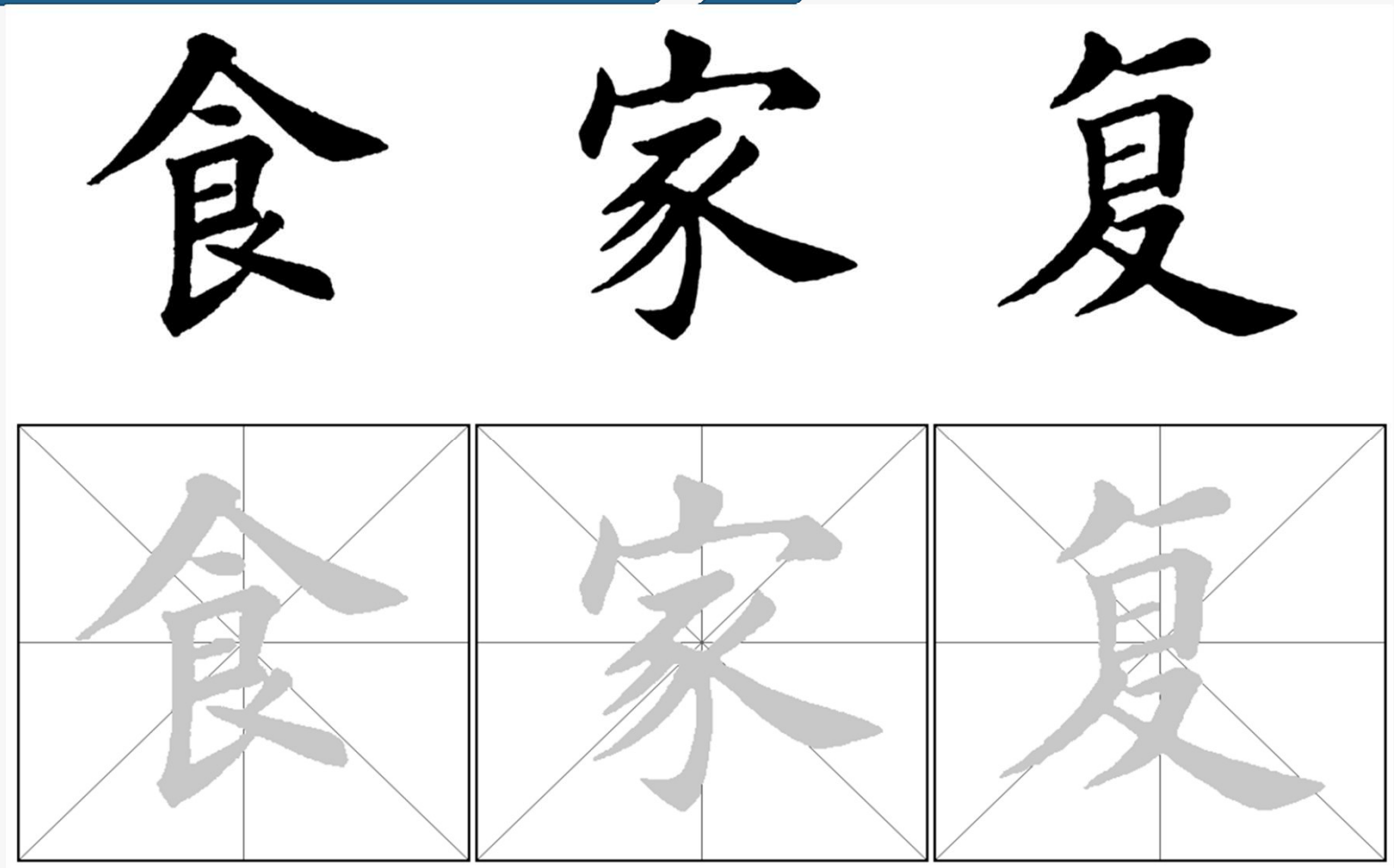
Key functions:

1. **Convert a color picture to grayscale**:
   "rgb2gray (variable)"
2. **Binarization of a picture**:
   1) Get the appropriate threshold by the method of maximum variance between classes:
      "graythresh (variable)"
   2) Use this threshold to binarize a picture:
      "im2bw (variable, threshold)"

Graphic-based Preparation



Graphic-based Preparation



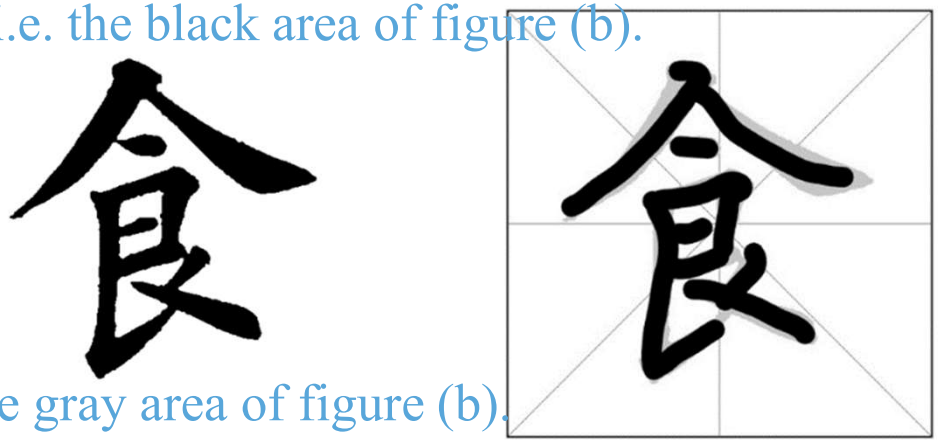
Each single standard Chinese character picture:
1080px × 1080px

Each copybook picture as background:
361px × 361px

Algorithm Based on Filling rate —— Theoretical Overview

Statistical targets:

1) “s”:
   The number of pixel in the standard character, i.e. the black area of figure (a).
2) “g”:
   The number of pixel in the testing character, i.e. the black area of figure (b).
3) “bingo”:
   The number of pixel in the overlapping part.
4) “more”:
   The number of pixel in the redundant part.
5) “less”:
   The number of pixel in the vacant part, i.e. the gray area of figure (b).

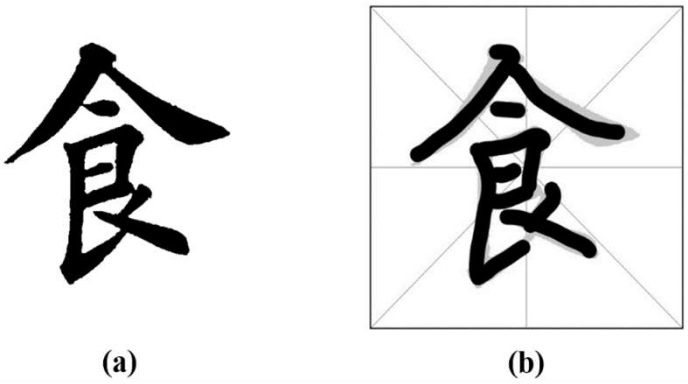


(a) (b)

bingo + less = s; bingo + more = g.

Algorithm Based on Filling rate —— Sample Test

1) Score_0 = 100 * bingo / s
2) Score_1 = 100 * (bingo – more) / s
3) Score_2 = 100 * (1 – more /g – less / s)



(a) (b)

	s	g	bingo	more	less	score_0	score_1	score_2
"食"	34368	30578	25933	4645	8435	75.4568	61.9413	60.2662
Test_1	10000	10000	0	10000	10000	0	-100	-100
Test_2	10000	200	0	200	10000	0	-2	-100
Test_3	10000	0	0	0	10000	0	0	--
Test_4	10000	30000	10000	20000	0	100	-100	33.3333

Algorithm Based on Filling rate —— Final Implementation

Get the values of three primary colors:

```
int pixel=mBitmap.getPixel(j,i);
int Red = (pixel & 0xff0000) >> 16;
int Green = (pixel & 0xff00) >> 8;
int Blue = (pixel &0xff);
```

Get its grayscale value through a famous visual formula:

```
grey = (int) (Red*0.299+Green*0.587+Blue*0.114)
```

Get two results and keep two decimals:

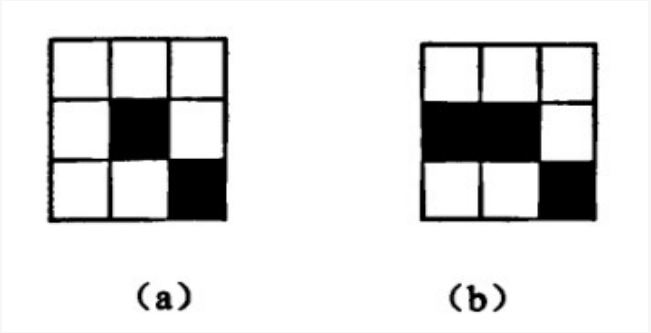
```
DecimalFormat df = new DecimalFormat("0.00");
score0 = df.format(100*bingo/(double)s);
score3 = df.format(Math.sqrt(100*(bingo-more/2)/(double)s)*10);
```

Algorithm for Skeletonization

Example of skeletonization of Chinese characters

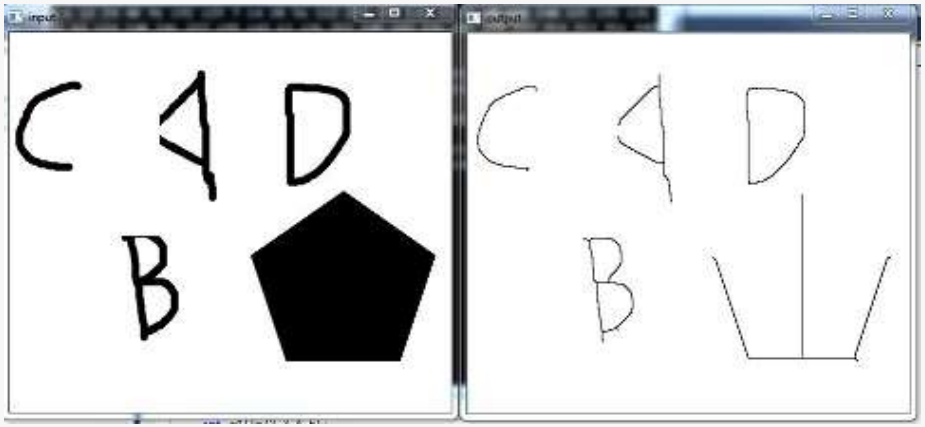
P3	P2	P1
P4	P	P0
P5	P6	P7

Eight adjacent pixels



Start point, end point, and other point

Algorithm for Skeletonization

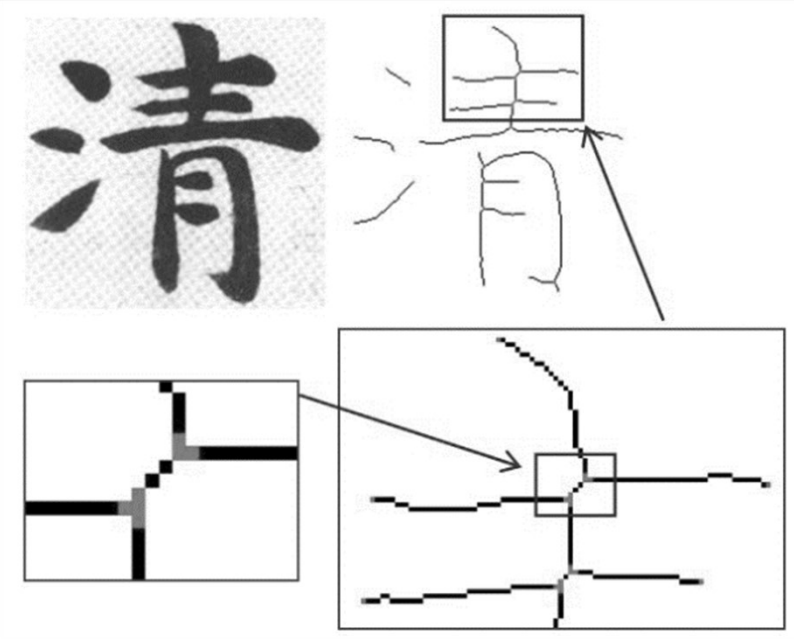


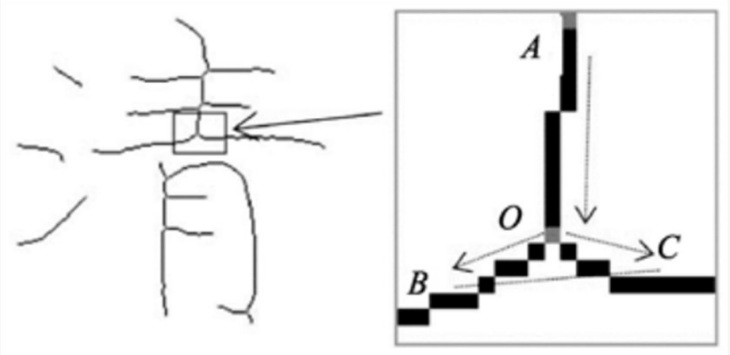
K3M algorithm, six-time check each iteration:

Phase 0: Mark the boundary of image.
Phase 1: If the point has 3 adjacent points, delete it.
Phase 2: If the point has 3 or 4 adjacent points, delete it.
Phase 3: If the point has 3, 4 or 5 adjacent points, delete it.
Phase 4: If the point has 3, 4, 5 or 6 adjacent points, delete it.
Phase 5: If the point has 3, 4, 5, 6 or 7 adjacent points, delete it.
Phase 6: Cancel the rest marked points, if there are no points being changed in phase 5, stop iteration, or back to phase 0 to start a new round.

Extracting Strokes

1. one-pixel wide skeletons of a calligraphic character are computed by corrosion.
2. stroke crawlers walk along these skeletons to obtain the moving trajectory. When meeting a cross, agent crawlers are sent to explore the potential right ways according to the writing rules.
3. After an individual stroke is obtained, its trajectory is coded using 8-directional chain code and used to compute stroke type. According to the stroke type, crossing area and corresponding contours are assigned to different strokes.
4. Finally, all the individual strokes with width of the same character are extracted.

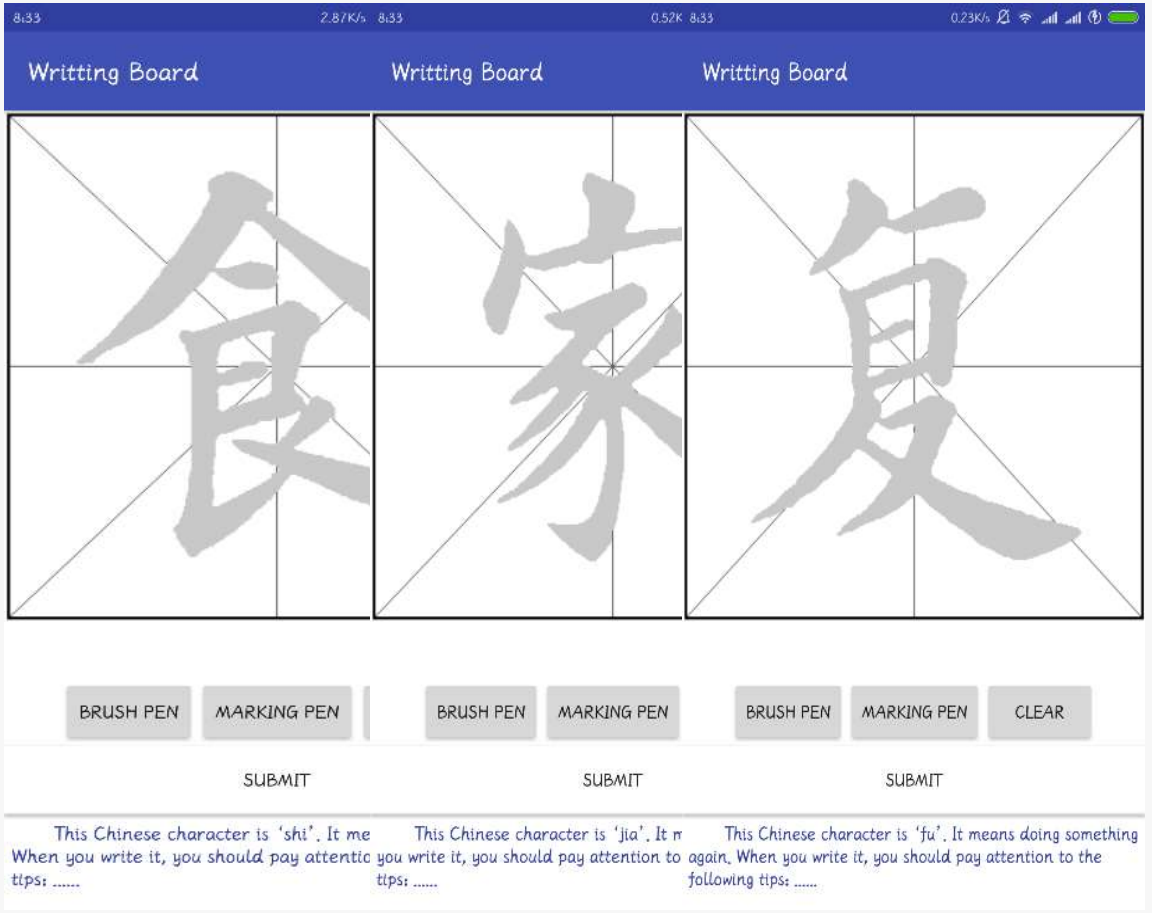




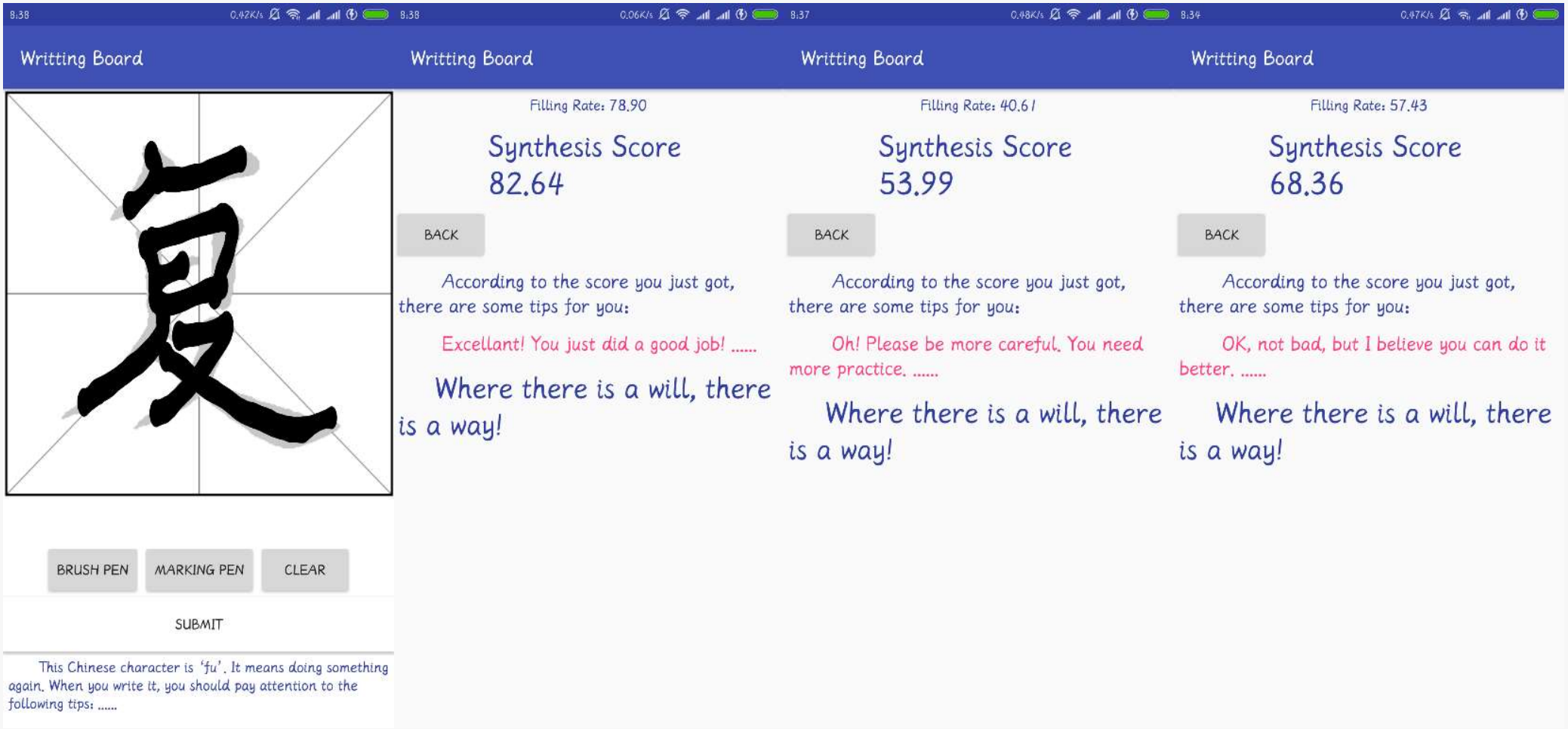
Algorithm Summary

Algorithm	Situation
Algorithm based on filling rate	Comparison of pixelated pictures without dislocation
Algorithm for Skeletonization	Allowing handwriting to be shifted and rotated.
Extracting Strokes	Adding score points to the details of different strokes.

User Interface



Using Effect



Future Work

1) Details optimization:

- ✓ Brush strokes
- ✓ User Interface

2) Evaluation methodology:

- ✓ More complex algorithm implementation
- ✓ Three aspects:

  overall evaluation, partial scoring, timekeeping

3) Feature database of characters and strokes


Thank you !
By Gu Jiapan

01 **Introduction**

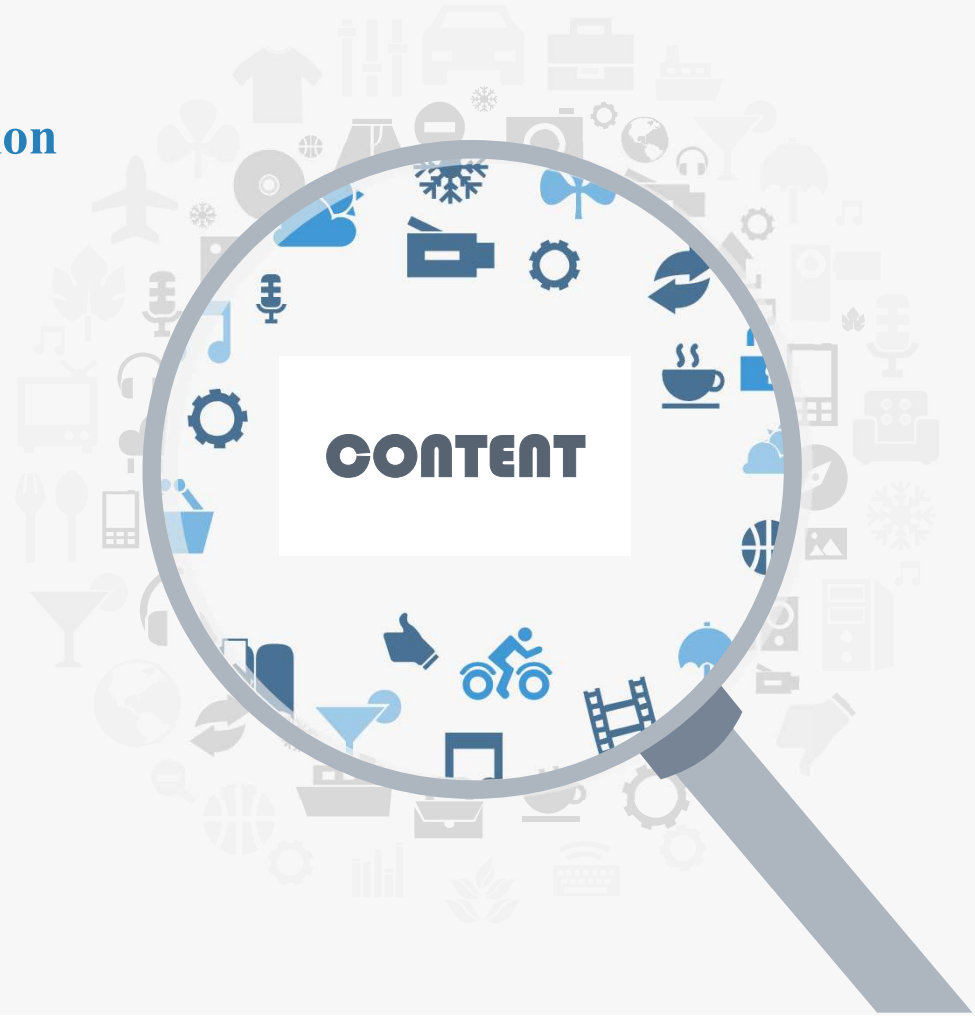
02 **Android Development Foundation**

03 **Application Prototype (Based on Writing board)**

04 **Algorithm of Evaluation**

05 **Application Display**

06 **Future Work**

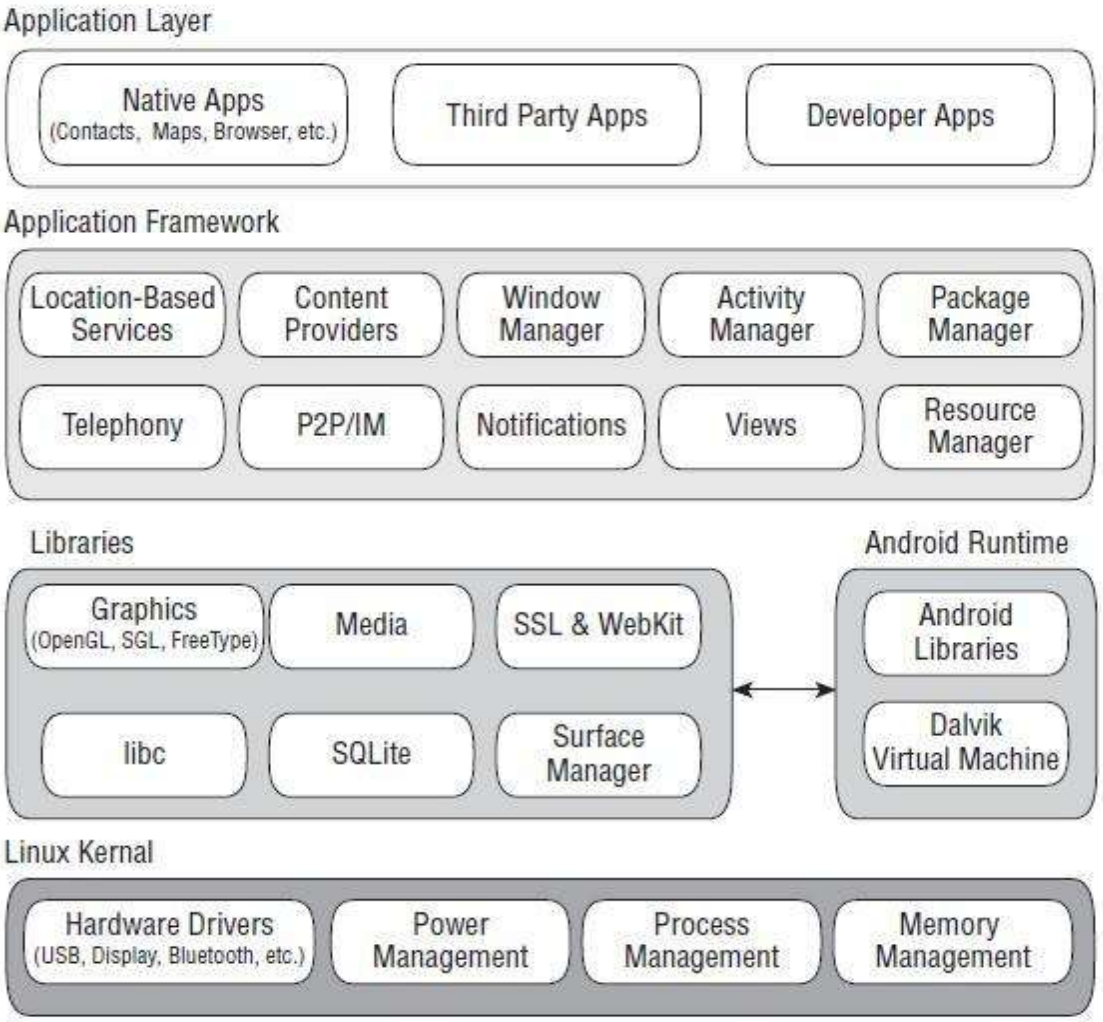


Chapter 2

Android development foundation

- Systematic Framework
- Version History
- Application Development Feature
- Software and Environment Configuration
- Program Project Structure

Systematic Framework



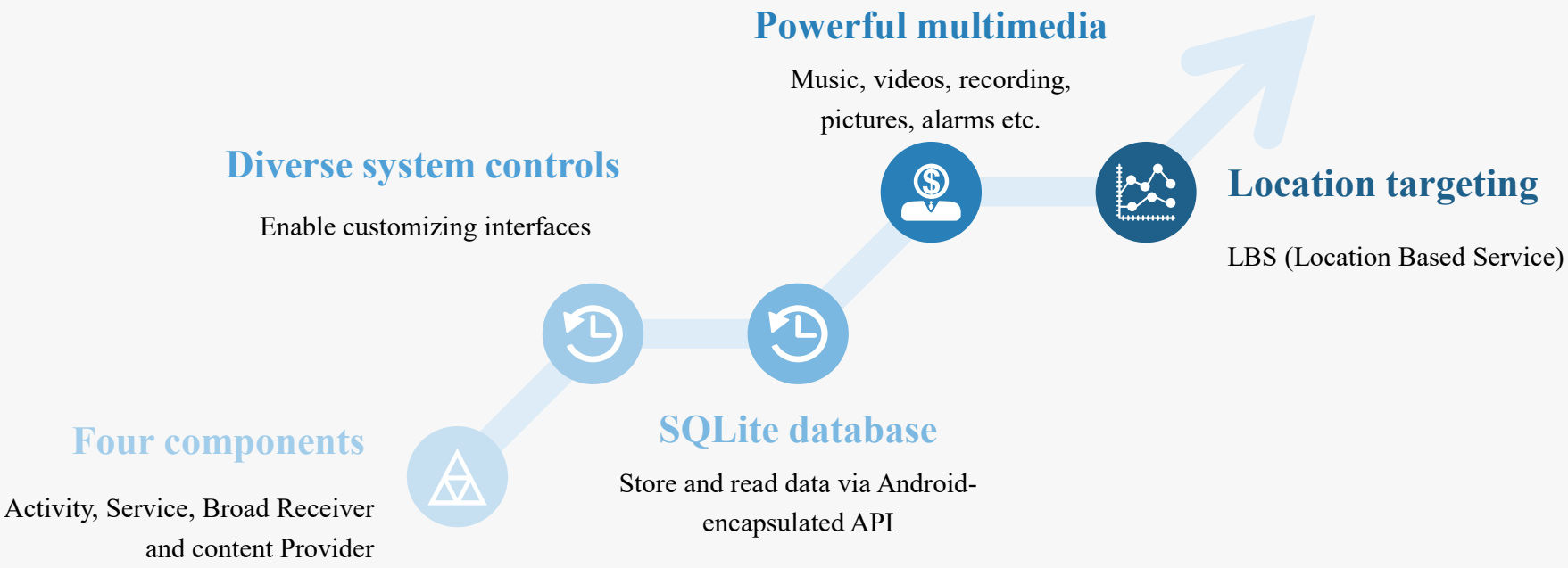
Version History

Code name	Version number	Initial release date	API level	Security patches[1]
(No codename)[2]	1.0	September 23, 2008	1	Unsupported
(Internally known as "Petit Four")[2]	1.1	February 9, 2009	2	Unsupported
Cupcake	1.5	April 27, 2009	3	Unsupported
Donut[3]	1.6	September 15, 2009	4	Unsupported
Eclair[4]	2.0 – 2.1	October 26, 2009	5 – 7	Unsupported
Froyo[5]	2.2 – 2.2.3	May 20, 2010	8	Unsupported
Gingerbread[6]	2.3 – 2.3.7	December 6, 2010	9 – 10	Unsupported
Honeycomb[7]	3.0 – 3.2.6	February 22, 2011	11 – 13	Unsupported
Ice Cream Sandwich[8]	4.0 – 4.0.4	October 18, 2011	14 – 15	Unsupported
Jelly Bean[9]	4.1 – 4.3.1	July 9, 2012	16 – 18	Unsupported
KitKat[10]	4.4 – 4.4.4	October 31, 2013	19 – 20	Unsupported[11]
Lollipop[12]	5.0 – 5.1.1	November 12, 2014	21 – 22	Unsupported[13]
Marshmallow[14]	6.0 – 6.0.1	October 5, 2015	23	Supported
Nougat[15]	7.0 – 7.1.2	August 22, 2016	24 – 25	Supported
Oreo[16]	**8.0 – 8.1**	August 21, 2017	26 – 27	Supported
Android P	9			Developer preview; not yet supported

Legend: Old version | Older version, still supported | **Latest version** | Latest preview version

- ✓ Latest version: 8.0
  Supported but not too stable
- ✓ Program project application:
  After 4.0 (almost 100% users)
- ✓ Actual trial: 7.1
  Great popularity

Application Development Feature



Software and Environment Configuration

1) JDK:

- ✓ Private JVM and resources
- ✓ Current version: Java SE 8.0 (1.8.0)

2) Android SDK:

- ✓ Provided by Google
- ✓ Relevant API
- ✓ Corresponds to Android system

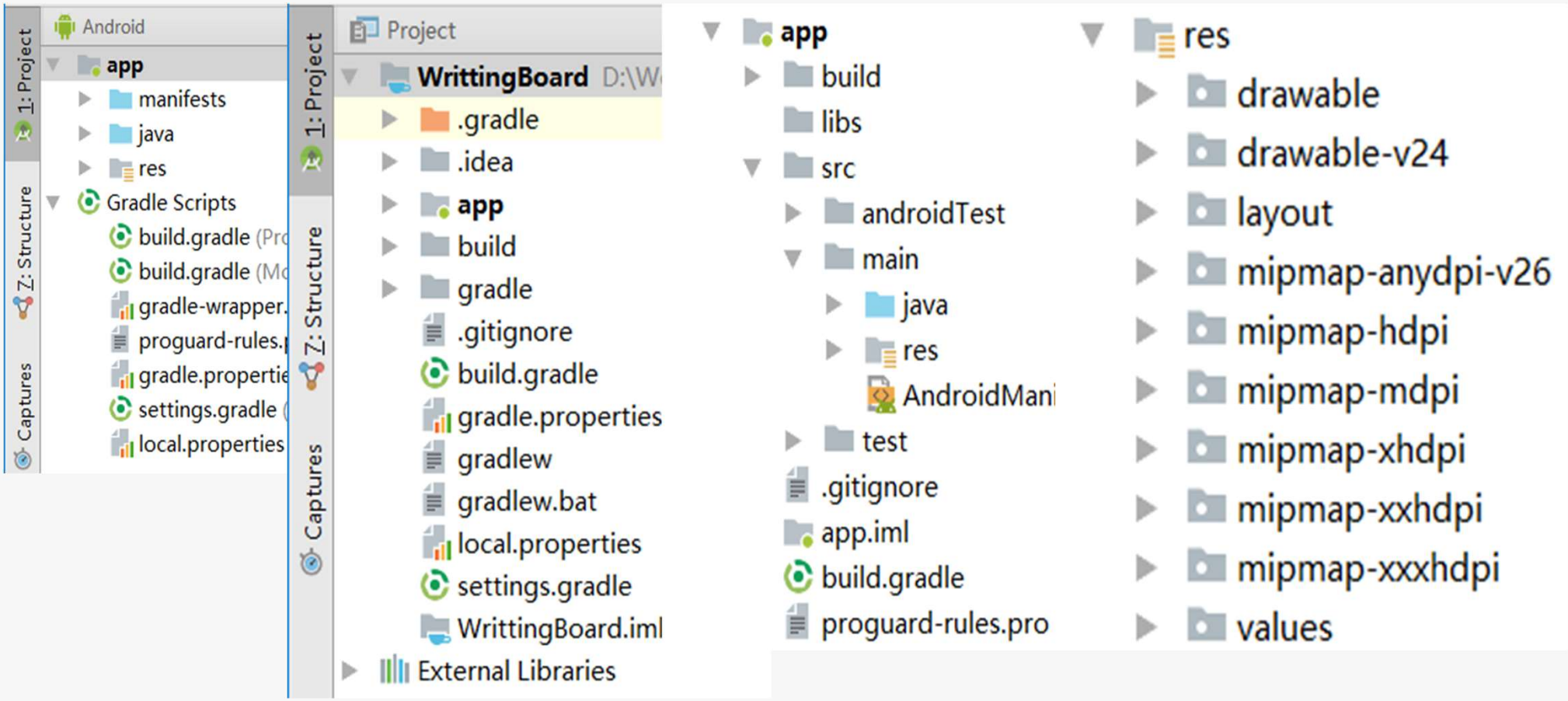
3) Android Studio:

- ✓ Official IDE tool
- ✓ More powerful than Eclipse
- ✓ Current version: 2.2

4) Other software:

- ✓ Eclipse:
  Convenient practice for Java and algorithm test
- ✓ MATLAB:
  Picture preprocessing and sample test

Program Project Structure



Activity Description —— Life Cycle