

Homework 2 -Basic Calculator

Kuan-Ting Lai 2019/3/20

Building a Basic Calculator

- Create a basic calculator using Java Swing
- Implement basic functions:
 - Addition (+)
 - Subtraction ()
 - Multiplication (*)
 - Division (/)
 - Minus-plus (±)
 - Clear (C)
 - Cancel Entry (CE)

🛓 Calcu	ulat —		×
CE	Clear		1
7	8	9	*
4	5	6	-
1	2	3	+
±	0		=

Building Calculator using IntelliJ

• Install IntelliJ Community

JET BRAINS	Tools	Language	es Soluti	ons	Support	Store	e a
IntelliJ IDEA	Coming	in 2019.1	What's New	Features	s Learn	Buy	Download

IJ	Windows macOS Linux	Sownload Intellij IDEA Windows macOS Linux		
	Ultimate	Community		
Version: 2018.3.5	For web and enterprise development	For JVM and Android development		
Build: 183.5912.21 Released: February 26, 2019 <mark>Release notes</mark>	DOWNLOAD .EXE	DOWNLOAD .EXE		
System requirements Installation Instructions Previous versions	Free trial	Free, open-source		
License	Commercial	Open-source, Apache 2.0 ?		
Java, Kotlin, Groovy, Scala	✓	×		

3

Create a Java Project

- Press Next
- Don't select any template



Add GUI Form

📱 Calculator [C:\Users\User\IdeaProjects\Calculator] - IntelliJ IDEA



Right Click

1	Calculator >	c 👌 Ċ CalculatorUI			
ect	Project 🔻	🕀 🚠 🏟 -	-		
Proj	Calculator	C:\Users\User\IdeaProjects`	\Ca		
÷	> 📄 .idea				
	> 📄 out				
	> 📄 src	New			
	Calculato	New	7		
	> IIII External Libr	X Cu <u>t</u>	Ctrl+X		
	📉 Scratches an	∎ <u>С</u> ору	Ctrl+C	Caratab File	
		C <u>o</u> py Path	Ctrl+Shift+C		
		Cop <u>y</u> Reference	Ctrl+Alt+Shift+C		
		Paste	Ctrl+V		
		Find <u>U</u> sages	Alt+F7	package-into.java	
		Find in <u>P</u> ath	Ctrl+Shift+F	module-Info.java	
		Repl <u>a</u> ce in Path	Ctrl+Shift+R	HTML File	e Doi
		Analy <u>z</u> e	>	🤄 .editorconfig file	.:£+ . N
		<u>R</u> efactor	>	Kotlin Script	IIIt+r
		Add to F <u>a</u> vorites	>	JavaFXApplication	E
		Show Image Thumbn	ails Ctrl+Shift+T	Singleton	-
		Reformat Code	Ctrl+Alt+L	G Gradle Kotlin DSL Build Script	t+Ho
		– Optimize Imports	Ctrl+Alt+O	G Gradie Kotlin DSL Settings	
		Delete	Delete	XSLI Stylesheet	oper
		Build Module 'Calcula	ator'	Edit File Templates	
		Rebuild ' <defaults'< td=""><td>Ctrl+Shift+F9</td><td>🚦 GUI Form</td><td></td></defaults'<>	Ctrl+Shift+F9	🚦 GUI Form	
			Carronnerro	🚦 Dialog	
		Show in Explorer		Form Snapshot	
	Run: Calcula	Open in Terminal Due Katlie Cental	Ctul , Alt. JAI	Resource Bundle	
	"C:\P	Kun Kotiin Scratch	Ctri+Alt+W	Plugin DevKit >	Brair
	• T	Local <u>H</u> istory	>		
	Draca	😋 Svnchronize 'src'			

Name Your Form "CalculatorForm"

💾 New GUI Form		\times
Form name:	CalculatorForm	$\uparrow \downarrow$
Base layout manager:	GridLayoutManager (IntelliJ) ~	
✓ Create <u>b</u> ound class		
Class name:	CalculatorForm	
	OK Cance	I

Set JPanel's Name

• Select the JPanel in the Component tree of the form view and update the field name property to calculatorView.



Generate main() Code

• In the code editor of Calculator.java file select -> Generate... -> Form main()



Run Main()



Create Action Listeners of Buttons



Enter Your Code in ActionListener

```
public class CalculatorForm {
    private JTextField displayField;
    private JPanel CalcPanel;
    private JButton buttonCE;
    private JButton button0;
    .....
    .....
    public CalculatorForm() {
        button0.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
Enter Your Own Code Here
        });
     .....
```

Two Modes in Calculator

Entering digits

Show results (temporary or final)

Example		
123 +	456	
Show temp result	579	
	579	- 789

Calculator			
≡ St	andard		U
		123	+ 456 +
			579
			0.0
MC	MR M+	M- N	ls M*
%	\checkmark	<i>x</i> ²	1/x
CE	С	$\langle X \rangle$	<u>.</u>
7	8	9	×
4	5	6	_
1	2	3	+
±	0		=

Define Variables

- Use enum to define operations
- Other variables
 - Mode (isDigitEnterMode)
 - Current display (displayString)
 - Temporary result (result)
 - Last operation (lastOP)

```
public class CalculatorForm {
    .....
    enum CalcOP {NONE, ADD, SUB, MULTIPLY, DIVIDE};
    private boolean isDigitEnterMode = false;
    private String displayString = "";
    private double result = 0;
    private CalcOP lastOP = CalcOP.NONE;
    .....
```

Adding Functions to Listeners of Digit Buttons

```
button0.addActionListener(new ActionListener() {
   @Override
    public void actionPerformed(ActionEvent e) {
        enterDigit("0");
    }
});
button1.addActionListener(new ActionListener() {
   @Override
    public void actionPerformed(ActionEvent e) {
        enterDigit("1");
    }
});
button2.addActionListener(new ActionListener() {
   @Override
    public void actionPerformed(ActionEvent e) {
        enterDigit("2");
    }
});
```

Entering Digits

• Call enterDigit() in each listener of digit buttons

```
private void enterDigit(String digit)
ł
    if (!isDigitEnterMode) {
        if (digit == ".")
            displayString = "0.";
        else
            displayString = digit;
        isDigitEnterMode = true;
    else {
        // Only floating-point number
        // can start with 0
        if (displayString == "0" && digit != ".")
            return;
        displayString += digit;
    displayField.setText(displayString);
```

Adding Functions to Listeners of OP Buttons

```
. . . . . . . .
buttonMultiply.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        evalLastOP(CalcOP.MULTIPLY);
});
buttonDivide.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        evalLastOP(CalcOP.DIVIDE);
});
buttonEqual.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        evalLastOP(CalcOP.NONE);
});
```

Evaluate Operations

Evalute operators (+ - * / =)

ť

}

```
private void evalLastOP(CalcOP currOP)
    double value = Double.parseDouble(displayField.getText());
    // Note that we evaluate last Operator, not current
    switch (lastOP) {
            case ADD:
                result += value;
                break;
            case SUB:
                result -= value;
                break;
            case DIVIDE:
                result /= value;
                break:
            case MULTIPLY:
                result *= value;
                break;
            default: // First value
                result = value;
                break;
    displayField.setText(Double.toString(result));
    isDigitEnterMode = false;
    lastOP = currOP;
```

Create Test Interfaces

- Need to provide two public test interfaces:
- 1. public void testClick(String button)
 throws Exception

```
2. public double getResult() {
    return result;
  }
```

3. public void showWindow()

public void testClick(String button) throws Exception

switch (button)

ł

}

```
case "+": buttonAdd.doClick(); break;
case "-": buttonSub.doClick(); break;
case "*": buttonMultiply.doClick(); break;
case "/": buttonDivide.doClick(); break;
case ".": buttonDot.doClick(); break;
case "=": buttonDivide.doClick(); break;
case "±": buttonMinusPlus.doClick(); break;
case "CE": buttonCE.doClick(); break;
case "CLEAR": buttonCLS.doClick(); break;
case "0": button0.doClick(); break;
case "1": button1.doClick(); break;
case "2": button2.doClick(); break;
case "3": button3.doClick(); break;
case "4": button4.doClick(); break;
case "5": button5.doClick(); break;
case "6": button6.doClick(); break;
case "7": button7.doClick(); break;
case "8": button8.doClick(); break;
case "9": button9.doClick(); break;
default:
```

throw new Exception("Error! No button " + button);

Show the Testing Process

Define a public function public void showWindow()

```
public void showWindow() {
    JFrame frame = new JFrame("Calculator");
    frame.setContentPane(this.CalcPanel);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.pack();
    frame.setVisible(true);
}
```

Build a Jar

 File -> Project Structure -> Project Settings -> Artifacts -> Click green plus sign -> Jar -> From modules with dependencies...

Project Structure		
$\leftarrow \ \rightarrow$	+ - 1	
Project Settings	Add	
Modules	Android Application	modules with dependencies
Libraries	JavaFx Application P JavaFx Preloader	
Facets	💠 Other	
Artifacts Platform Settings		
SDKs		
Global Libraries		
Problems	NUCL N	
Troberts	Nothing to show	

Build a Jar (Cont'd)

• Build | Build Artifact

型 Calculator [C:\Users\User\OneDrive\Teaching\107-2物件導向程式設計\demo\Calculator] - ...\src\CalculatorForm.java [Calculator] - IntelliJ IDEA

Testing Your Calculator with JUnit

- Download junit-4.12.jar & hamcrest-core-1.3.jar
- Download CalculatorFormTest.java
- Compile CalculatorFormTest.java with your jar
 C:\> javac -cp ".;junit-4.12.jar;Calculator.jar" CalculatorFormTest.java
- Run the Test

C:\> java -cp ".;junit-4.12.jar;Calculator.jar" CalculatorFormTest

Testing Process

Create Unit Testing using IntelliJ

Select Your Class and Press "Alt + Enter"

🖳 Calculator [C:\Users\User\OneDrive\Teaching\107-2物件導向程式設計\homework\hw2\Calculator]\src\CalculatorForm.java [Calculator] - IntelliJ IDEA 🛛 🗖				\times
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>N</u> avigate <u>C</u> ode Analy <u>z</u> e <u>R</u> efa	ctor <u>B</u> ui	d R <u>u</u> n <u>T</u> ools VC <u>S</u> <u>W</u> indow <u>H</u> elp		
Calculator > Src > CalculatorForm		🕞 🔨 🗐 CalculatorFormTest 🗸 🕨 🝎 🕻	G ■	Q 📭
ਹੂ 📄 Project 🔻 😳 😤 🗱 🗕	Calcul	atorForm.java × CalculatorFormTest.java × 🗄 CalculatorForm.form ×		*
🖁 🗸 🖿 Calculator C:\Users\User\OneDrive\Teac	1	import		Ant
🛗 🗲 🖿 .idea	5			Bui
▼ ■ out	6 🕨 🛔	<pre>public class CalculatorForm {</pre>		E
✓ artifacts	7 1	private JTextFiel Run 'CalculatorForm.main()' Ctrl+Shift+F10		m
Calculator iar	8	private JPanel Ca to Debug 'CalculatorForm.main()'		Z
	10 OK	private JButton b Run 'CalculatorForm.main()' with Coverage		aver
> production	10 OK	private JButton b 7 Create Test		
> test	12 OK	private JButton b I Create subclass		
	13 ОК	private JButton b 求 Add Javadoc 🕒		
	14 ОК	private JButton b 求 Make package-private		_
	15 ок	private JButton button5;		
	16 OK	private JButton button6;		
	18 OK	private JButton button8:		
	19 OK	private JButton button9:		
	20 OK	private JButton buttonEqual;		
	21 ОК	private JButton buttonAdd;		
IIII External Libraries	22 ОК	<pre>private JButton buttonMultiply;</pre>		
hamcrest-core-1.3.jar library root	23 ОК	private JButton buttonSub;		
junit-4.12.jar library root	24 ок	private JButton buttonDivide;		
Image: Antiperiod Action and Antiperiod Action and A	25 OK	private JButton buttonDot;		-
JUnit4	27 06	private Jutton buttonCLS:		
JUnit5.3	28	private souton battonets,		
> Til testng	29	<pre>enum CalcOP {NONE, ADD, SUB, MULTIPLY, DIVIDE};</pre>		
Scratches and Consoles	30			

Select Your Test Framework

• We use JUnit4 here.

🖳 Create Test		×
Testing library:	♣ JUnit4	\sim
Class name:	CalculatorFormTest	
Superclass:		~
Destination package:		~
Generate:	☐ set <u>U</u> p/@Before ☐ tear <u>D</u> own/@After	
Generate test methods for:	Show <u>i</u> nherited methods	
	Member	
🗌 💼 🕤 testClick(button:	String):void	
🗌 ៣ 🖢 🛛 getResult():doub	le	
🗌 🧰 🕤 showWindow():ve	bid	
🗌 🔊 🖕 main(args:String)]):void	
?	OK	Cancel

Run Your Test

• Press "Alt + Tab" on your test class

🖳 Calculator [C:\Users\User\OneDrive\Teaching\107-2物件導向程式設計\homework\hw2\Calculator] - ...\src\CalculatorFormTest.java [Calculator] - IntelliJ IDEA Ē X <u>File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help</u> **Calculator** > **src** > **CalculatorFormTest** CalculatorFormTest ~ **益** Q **F** CalculatorFormTest.java > 🗧 CalculatorForm.form Project \oplus \div CalculatorForm.java Project 💌 ÷ Ant Build Calculator C:\Users\User\OneDrive\Teac 1 import org.junit.Test; import static org.junit.Assert.*; 2 > idea 3 import org.junit.runner.JUnitCore; out import org.junit.runner.Result; 4 ✓ artifacts m 5 import org.junit.runner.notification.Failure; Maven Calculator_jar 6 Calculator.jar 7 8 public class CalculatorFormTest { > production 9 Ctrl+Shift+F10 Run 'CalculatorForm....main()' > test static CalculatorF 🛓 Debug 'CalculatorForm....main()' 10 ✓ src 11 🕼 Run 'CalculatorForm....main()' with Coverage > META-INF private double eva 12 @ Ctrl+Shift+F10 CalculatorFormTest double result 13 **İ** Debug 'CalculatorFormTest' char [] in arr 14 Run 'CalculatorFormTest' with Coverage 15 trv { CalculatorForm for (char | reate subclass 16 CalculatorForm.form if (c | 17 Add Javadoc Calculator.iml 18 Make package-private External Libraries 19 hamcrest-core-1.3.jar library root > calc.testClick(Character.toString(c)); 20 **)** junit-4.12.jar library root 21 22 result = calc.getResult(); Equation C:\Program Files\Java\jdk-11.0 23 JUnit4 > 24 catch (Exception e) { 11 ILLINE 2