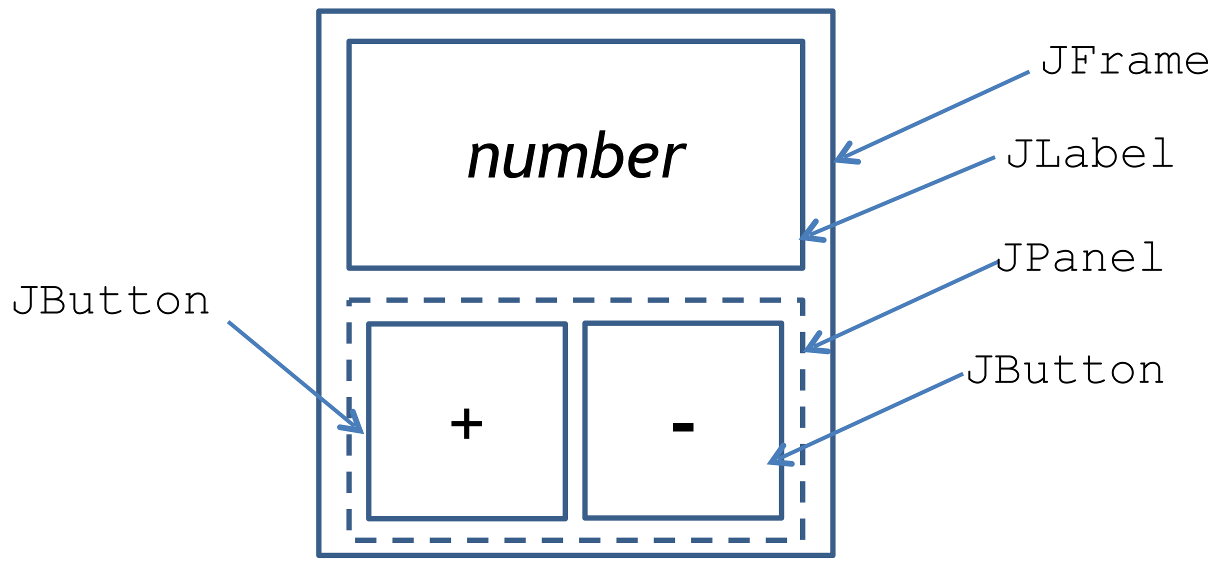
Software Development 2

Lecture 5: Dynamic interfaces

Today’s Lecture

- Program structure
- Editable text
- Dynamically changing the interface

Example: counter



- display *number* in `JLabel`
- `JButton`s to increment/decrement number

Example: counter

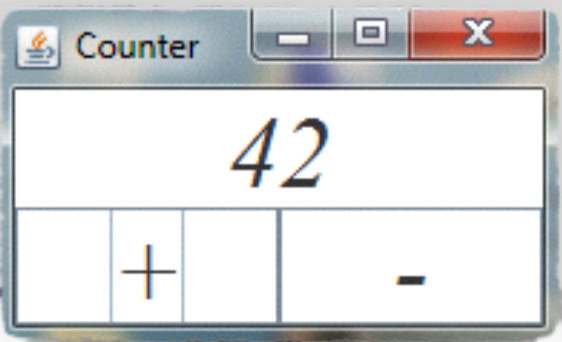
```
class Counter
 extends JFrame implements ActionListener
{   JLabel output;    // counter output
    JButton up,down;  // + and - buttons
    JPanel p;         // button panel
    int value = 0;    // state variable

    public Counter()
    {   setLayout(new GridLayout(2,1));
        Font f = new Font("Serif",Font.ITALIC,36);
        output = new JLabel("0",JLabel.CENTER);
        output.setFont(f);
        add(output);
```

Example: counter

```
    p = new JPanel(new GridLayout(1,2));

    up = new JButton("+");
    up.setFont(f);
    up.setBackground(Color.white);
    p.add(up);
    // The containing counter object handles button events
    up.addActionListener(this);

    down = new JButton("-");
    down.setFont(f);
    down.setBackground(Color.white);
    p.add(down);
    down.addActionListener(this);

    add(p); // add button panel to bottom of frame
  }
```

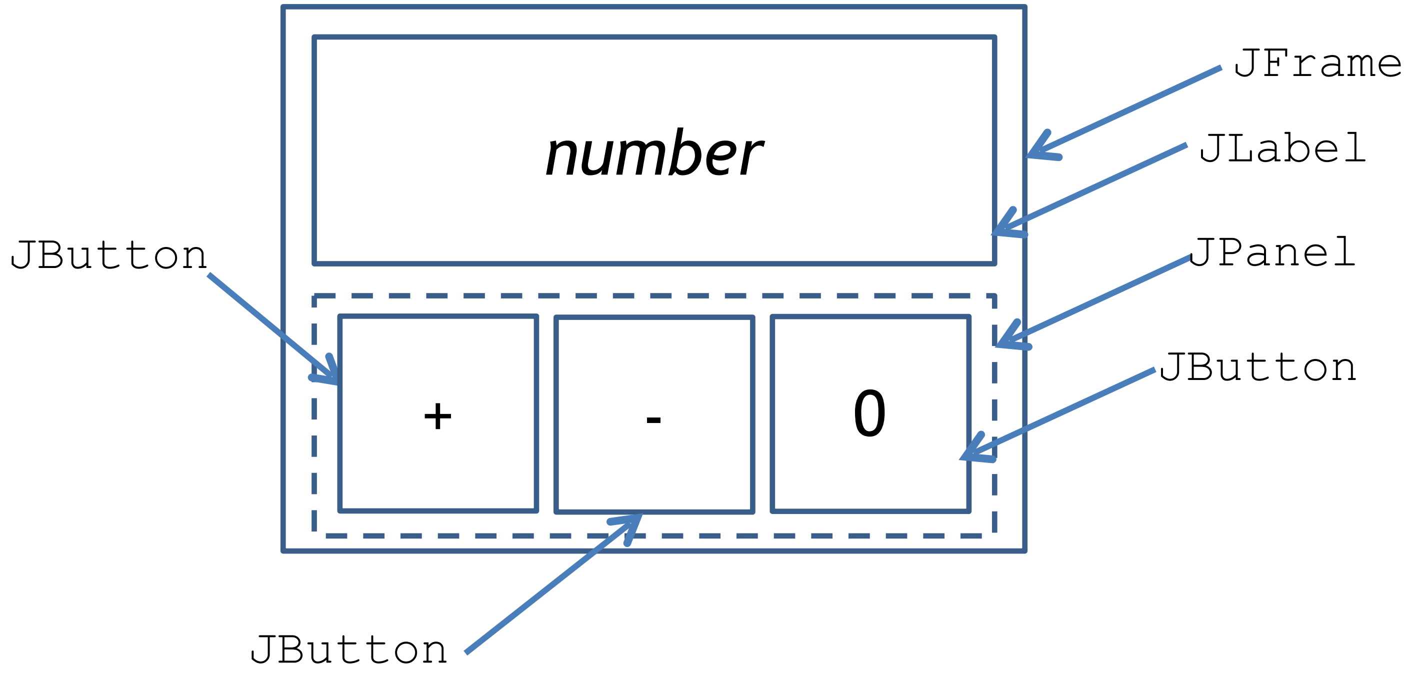
Example: counter

```
   // handle events from buttons
   public void actionPerformed(ActionEvent e)
   {  if(e.getSource()==up) // + button pressed
         value++;
      else
      if(e.getSource()==down) // - button pressed
          value--;
      output.setText(value+"");
   }
}

class TestCounter
{  ... }
```



Example: counter 2

- add `JButton` to reset number to 0



Example: counter 2

```
class Counter2
 extends JFrame implements ActionListener
{   JLabel output;
    JButton up,down,zero; // +, - and 0 buttons
    JPanel p;
    int value = 0;

    public Counter2()
    {   setLayout(new GridLayout(2,1));
        Font f = new Font("Serif",Font.ITALIC,18);

        output = new JLabel("0",JLabel.CENTER);
        output.setFont(f);
        output.setBackground(Color.white);
        add(output);
```

Example: counter 2

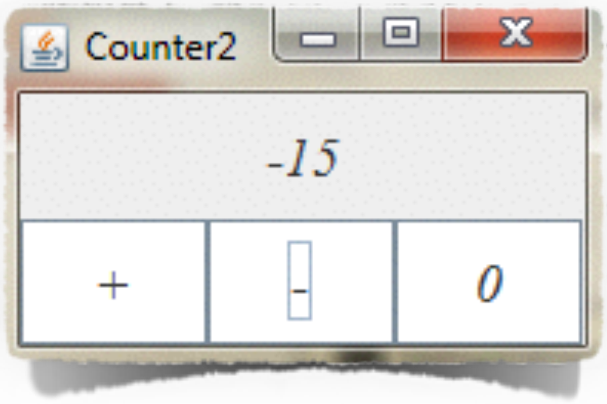
```
p = new JPanel(new GridLayout(1,3));

up = new JButton("+");
...

down = new JButton("-");
...

// create zero button
zero = new JButton("0");
zero.setFont(f);
output.setBackground(Color.white);
p.add(zero);
zero.addActionListener(this);

add(p);
```

Example: counter 2

```
// handle events from buttons
public void actionPerformed(ActionEvent e)
   {  if(e.getSource()==up)
         value++;
      else
      if(e.getSource()==down)
         value--;
      else
      if(e.getSource()==zero) // 0 button pressed
         value=0;
      l.setText(value+"");
   }
}
```

Example: counter 2

```
class TestCounter2
{   ... }
```



Method or sub-class?

`Counter2` has lots of repeated code to set up `JButton`s

- generalise by constructing a new method:

```
class Counter3 extends JFrame implements ActionListener {
    ...
    // creates a button with a particular look and feel
    private JButton C3Button(String text){
        JButton b = new JButton(text);
        b.setFont(new Font("Serif",Font.ITALIC,36));
        b.setBackground(Color.white);
        return b;
    }
```

Method or sub-class?

```
public Counter3()
{  ...
   p = new JPanel(new GridLayout(1,3));
   up = C3Button("+");
   up.addActionListener(this);
   p.add(up);
   down = C3Button("-");
   down.addActionListener(this);
   p.add(down);
   zero = C3Button("0");
   zero.addActionListener(this);
   p.add(zero);
   add(p);
}
```

The constructor is now much less cluttered.

Method or sub-class?

Recognise that we are actually constructing a specialised variant of `JButton`

- instead of writing method, sub-class `JButton`

```
//a new button sub-class with a particular look and feel
class C4Button extends JButton {
     C4Button(String text) {
        super(text);
        setFont(new Font("Serif",Font.ITALIC,36));
        setBackground(Color.white);
     }
}
```

Method or sub-class?

- `C4Button` is a new class which inherits all properties of the super class `JButton`
- `super(`*`text`*`)` calls the super-class constructor `JButton(`*`text`*`)` to make a new `JButton` object
- `setFont` and `setBackground` now implicitly affect the new `JButton` object created by `super`

Method or sub-class?

We can now use a `C4Button` anywhere we can use a `JButton`:

```
class Counter4 extends JFrame implements ActionListener {
    JLabel output;
    C4Button up,down,zero;
    JPanel p;
    int value = 0;
```

Method or sub-class?

```
public Counter4() {
    setLayout(new GridLayout(2,1));

    output = new JLabel("0",JLabel.CENTER);
    output.setFont(new Font("Serif",Font.ITALIC,36));
    output.setBackground(Color.white);
    output.setOpaque(true);
    add(output);

    p = new JPanel(new GridLayout(1,3));
```

Method or sub-class?

```
        up = new C4Button("+");
        up.addActionListener(this);
        p.add(up);

        down = new C4Button("-");
        down.addActionListener(this);
        p.add(down);

        zero = new C4Button("0");
        zero.addActionListener(this);
        p.add(zero);

        add(p);
    }
```

Text Field

Often want to input arbitrary information as text via an interface.

- use a JTextField:

```
public class JTextField
         extends JTextComponent
```

- a bit like a `JLabel`
- displays a single line of optionally editable text

Text Field

```
JTextField(int columns)
JTextField(String text)
JTextField(String text, int columns)
```

- `columns` => set preferred text width
- `text` => set initial text

```
getText()
```

- **return text at any time**

```
setText(String text)
```

- change text

Text Field

- use mouse to select `JTextField` for text entry
- press return to cause: `ActionEvent`
- detected by: `ActionListener`
- handled by: `actionPerformed`

- So, events are handled in the same way as a `JButton`.

INTERMISSION

Room: F27SB

Question 1

Which of these is an example of “form follows function”?

A. The importance of aesthetics in GUI design.

B. That Java code should be attractive to read.

C. That GUI controls should reflect the internal state of a program.

D. That on-screen forms can be generated by Java code.

Question 1

Which of these is an example of “form follows function”?

A. The importance of aesthetics in GUI design.
B. That Java code should be attractive to read.
C. That GUI controls should reflect the internal state of a program.
D. That on-screen forms can be generated by Java code.

Question 2

When a JPanel with a BorderLayout is enlarged in both dimensions:

A. All regions grow in size by the same amount in both dimensions.
B. The central region becomes larger in both dimensions.
C. The height of the north and south regions increases.
D. The width of the east and west regions increases.

Question 2

When a JPanel with a BorderLayout is enlarged in both dimensions:

A. All regions grow in size by the same amount in both dimensions.

B. **The central region becomes larger in both dimensions.**

C. The height of the north and south regions increases.

D. The width of the east and west regions increases.

Question 3

<u>Assertion</u>: When a user clicks a JButton with the mouse, an interrupt is generated.

<u>Reason</u>: Because class JButton implements the ActionListener interface.

A. The assertion is **true**, the reason is **true**.

B. The assertion is **true**, the reason is **false**.

C. The assertion is **false**, the reason is **false**.

D. The assertion is **false**, the reason is **true**.

Question 3

Assertion: When a user clicks a JButton with the mouse, an interrupt is generated.

Reason: Because class JButton implements the ActionListener interface.

A. The assertion is **true**, the reason is **true**.

B. The assertion is true, the reason is false.

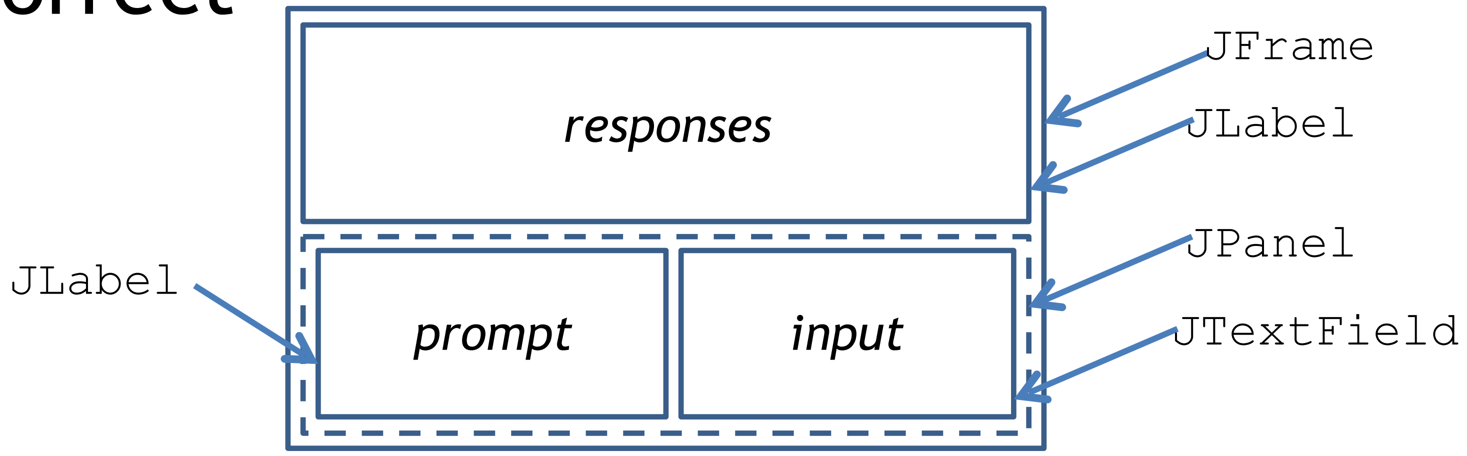
C. The assertion is **false**, the reason is **false**.

D. The assertion is **false**, the reason is **true**.

Some Examples

Example: number guessing

- Computer “thinks” of a number between 1 and 100
- User enters their guess as text
- Computer says if guess is high, low or correct



Example: number guessing

- use `BorderLayout` for `JFrame`
  - which is the default for a content pane
- add to north & centre
- use `FlowLayout` for `JPanel`
  - since *prompt* will be much bigger than *input*
  - which is the default for a JPanel

Example: number guessing

```
class Guess extends JFrame implements ActionListener {

    int number; //number being guessed
    JLabel response,prompt;
    JTextField input;
    JPanel p; //contains prompt and input

    //returns the absolute value of its argument
    int abs(int x) {
        if(x<0)
            return -x;
        return x;
    }
```

Example: number guessing

```
public Guess() {
    //get a random number between 1 and 100
    number = abs(new Random().nextInt())%100+1;

    Font f = new Font("serif",Font.ITALIC,18);
    response =
        new JLabel("I'm thinking of a
                   number between 1 and 100",
                   JLabel.CENTER);
    response.setFont(f);
    add(response, BorderLayout.NORTH);

    p = new JPanel();
```

Example: number guessing

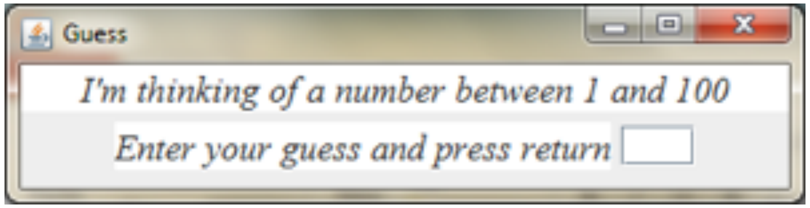
```
        prompt =
            new JLabel("Enter your guess and press return",
                       JLabel.CENTER);
        prompt.setFont(f);
        p.add(prompt);

        input = new JTextField(3);
        p.add(input);
        //add listener for action events from text field
        input.addActionListener(this);

        add(p, BorderLayout.CENTER);
    }
```

Example: number guessing

```
    public void actionPerformed(ActionEvent e) {
        if(e.getSource()==input) { //“enter” pressed
            //convert input string to int
            //and then print appropriate response
            int guess =
                Integer.parseInt(input.getText());
            if(guess==number)
                response.setText("Correct!");
            else
                if(guess<number)
                    response.setText("Low...");
            else
                response.setText("High...");
        }
    }
}
```

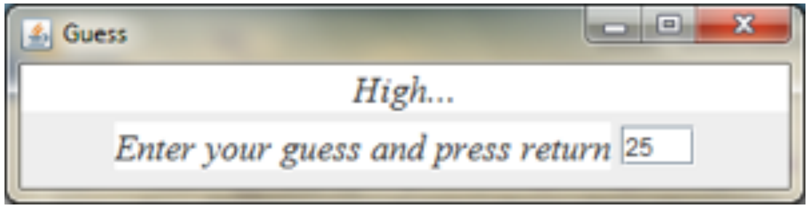
Example: number guessing

```
class TestGuess
{ ... }
```



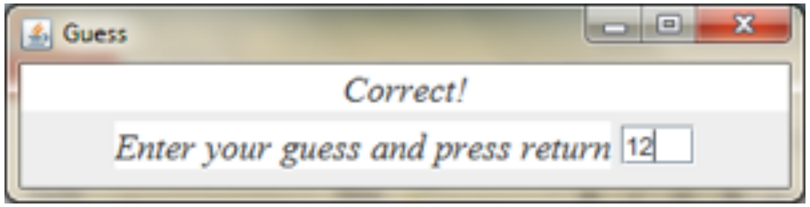
I'm thinking of a number between 1 and 100

Enter your guess and press return



High...

Enter your guess and press return 25



Correct!

Enter your guess and press return 12

DYNAMIC INTERFACE CHANGES

Dynamic interface changes

At the end of a guessing session offer options

- **play again** and **stop**

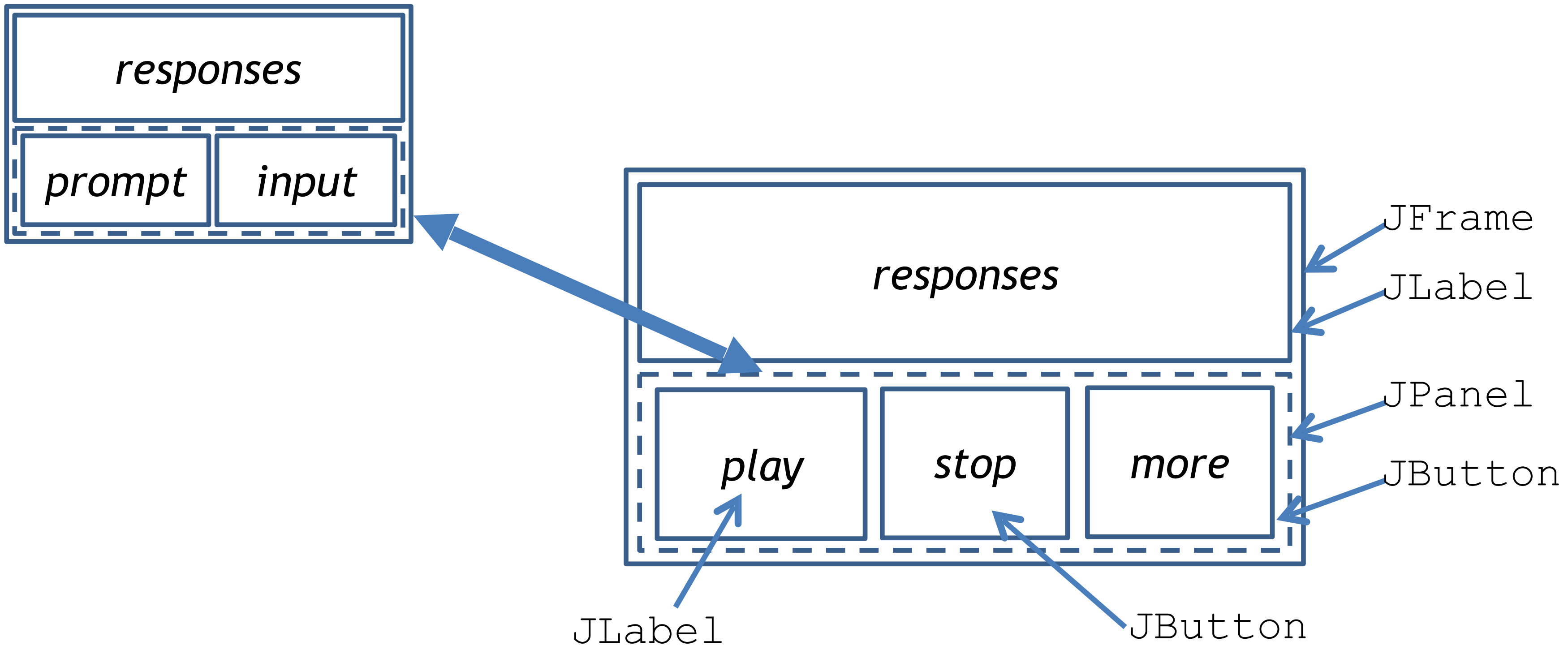
Introduce "more"/"stop" `JButtons`?

- add new `JButtons` to current `JPanel`
- disable `JButtons` during play
- enable `JButtons` and disable `JTextField` at end

- Danger of GUI becoming cluttered

Dynamic interface changes

Alternatively, we could **swap** current `JPanel` with new `JPanel` with the extra buttons



Dynamic interface changes

We can do this using:

`remove(`*`Component c`*`)`

- removes `Component` *`c`* from `Container`
- can then add new `Component` to `Container`
- Note: need to `setVisible(false)` for old `Component` and `setVisible(true)` for new `Component`

Dynamic interface changes

```
class Guess2 extends
 JFrame implements ActionListener
{   long number;
    JLabel response,prompt;
    JTextField input;
    JPanel p1,p2;
    JButton stop,more; //buttons for new panel p2
    JLabel play;       //message for new panel p2
    ...
```

Dynamic interface changes

```
// lots of similar labels and buttons, so add methods
JLabel setupLabel(String s,JPanel p)
{   JLabel l = new JLabel(s,JLabel.CENTER);
    l.setFont(new Font("serif",Font.ITALIC,18));
    p.add(l);
    return l;
}
// this method also adds an action listener to each button
JButton setupButton(String s,JPanel p)
{   JButton b = new JButton(s);
    b.setFont(new Font("serif",Font.ITALIC,18));
    p.add(b);
    b.addActionListener(this);
    return b;
}
```

Dynamic interface changes

```
public Guess2()
{   ...
    p2 = new JPanel();
    //create GUI elements, add to panel
    //and register button event listeners
    play = setupLabel("Play again?",p2);
    stop = setupButton("STOP",p2);
    more = setupButton("MORE",p2);
}
```

Dynamic interface changes

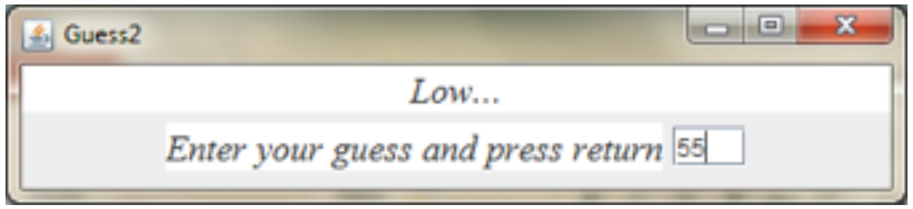
```
public void actionPerformed(ActionEvent e) {
  if(e.getSource()==input)
   {   int guess = Integer.parseInt(input.getText());
       if(guess==number)
       {   response.setText("Correct!");
            //swap guessing panel with button panel
           remove(p1);
           add(p2, BorderLayout.CENTER);
           p1.setVisible(false); p2.setVisible(true);
       }
       else
       if(guess<number)
           response.setText("Low...");
       else
           response.setText("High...");
   }
```

Dynamic interface changes

```
      // handle action events from the new buttons
      else
         if(e.getSource()==stop)
            System.exit(0);
      else
         if(e.getSource()==more) {
            // for more, choose a new number to guess
            // and then swap the panels back again
            remove(p2);
            add(BorderLayout.CENTER,p1);
            p2.setVisible(false); p1.setVisible(true);
            response.setText("I'm thinking of number between
                              1 and 100");
            number = abs(new Random().nextInt())%100+1;
            input.setText("");
      }
   }
}
```

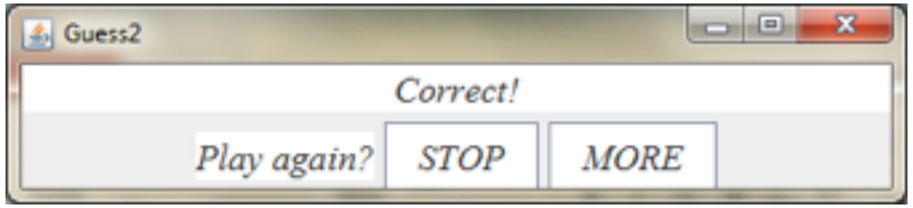
Dynamic interface changes

```
class TestGuess2
{   ... }
```



Low...

Enter your guess and press return 55

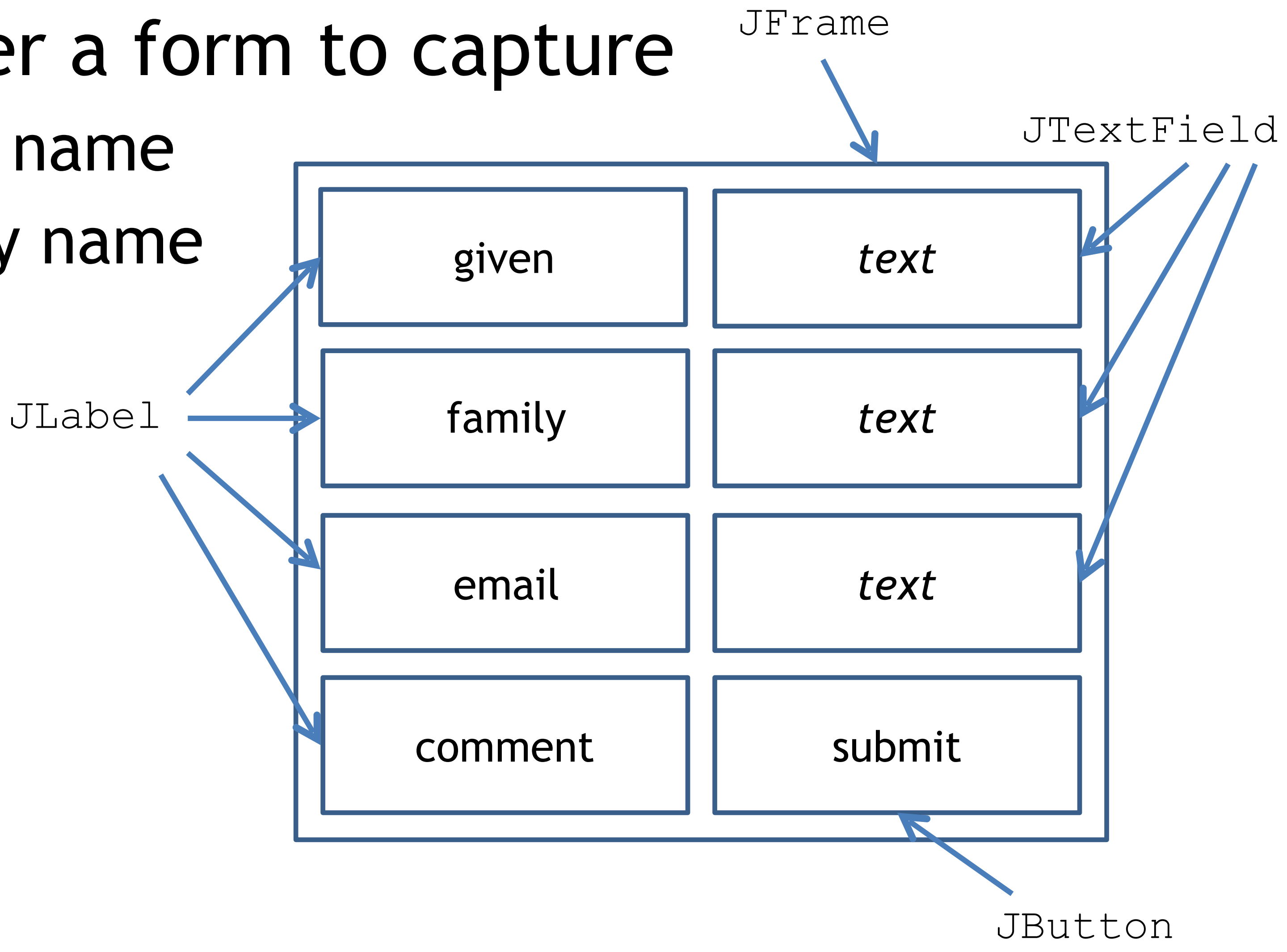


Correct!

Play again? STOP MORE

Example: form

- consider a form to capture
  - given name
  - family name
  - …



Example: form

When **submit** button is pressed:

- check all `JTextField`s have entries
- place any error message in *comment*
- write all entries to file
- swap *comment*/*submit* panel with *more*/*stop*

Example: form

Then when **more** pressed:

- swap panels back again
- clear all `JTextField`s

And when **stop** pressed:

- close the file
- exit the program

Example: form

```
class Form extends JFrame implements ActionListener
{
   JLabel given,family,email,comment; //fields
   JTextField gt,ft,et; //for user input
   JButton submit,more,stop; //buttons

   PrintWriter file; //for writing to a file
```

Example: form

New methods to set up `JLabel/JButton/ JTextField`

- also adds new object to specified `Container` *`c`*

```
JLabel setupLabel(String s, int style, Container c)
  {  JLabel l = new JLabel(s,JLabel.CENTER);
     l.setFont(new Font("serif",style,18));
     c.add(l);
     return l;
  }
```

Example: form

- Button method also adds an action listener:

```
JButton setupButton(String s, Container c)
{   JButton b = new JButton(s);
    b.setFont(new Font("serif",Font.ITALIC,18));
    c.add(b);
    b.addActionListener(this);
    return b;
}
```

Example: form

- Action events from text fields are ignored
  - Text will be read from all of them when submit is pressed

```
JTextField setupTextField(Container c)
{  JTextField t = new JTextField();
   t.setFont(new Font("sanserif",Font.PLAIN,18));
   c.add(t);
   return t;
}
```

Example: form

```
public Form()
{   setLayout(new GridLayout(4,2)); // form grid

    // set up and add the field labels and text fields
    given = setupLabel("Given name",Font.PLAIN,this);
    gt = setupTextField(this);
    family = setupLabel("Family name",Font.PLAIN,this);
    ft = setupTextField(this);
    email = setupLabel("Email address",Font.PLAIN,this);
    et = setupTextField(this);
    comment = setupLabel("",Font.ITALIC,this);

    // set up, add, and register listener of the button
    submit = setupButton("SUBMIT",this);
```

Example: form

```
    // also setup more and stop, but don’t yet add them!
    more = new JButton("MORE");
    more.setFont(new Font("serif",Font.ITALIC,18));
    more.addActionListener(this);

    stop = new JButton("STOP");
    stop.setFont(new Font("serif",Font.ITALIC,18));
    stop.addActionListener(this);
}
```

Example: form

```
// handle events from buttons
public void actionPerformed(ActionEvent e)
{   if(e.getSource()==submit)
     doSubmit();
    else
    if(e.getSource()==more)
     doMore();
    else
    if(e.getSource()==stop)
     doStop();
}
```

Example: form

```
// will be called when submit button is pressed
void doSubmit() {
   // make sure valid input has been entered
   if(gt.getText().equals(""))
      comment.setText("Enter given name");
   else
   if(ft.getText().equals(""))
    comment.setText("Enter family name");
   else
   if(et.getText().equals(""))
    comment.setText("Enter email address");
   else
   // if valid, then save input to disk file
   {  file.println(gt.getText());
      file.println(ft.getText());
      file.println(et.getText());
```

Example: form

```
        // and swap comment/submit with more/stop buttons
        remove(comment);
        comment.setVisible(false);
        remove(submit);
        submit.setVisible(false);
        add(more);
        more.setVisible(true);
        add(stop);
        stop.setVisible(true);
        setVisible(true);
    }
}
```

Example: form

```
// will be called when more button is pressed
void doMore() {
    // reset text input fields
    comment.setText(""); gt.setText("");
    ft.setText(""); et.setText("");
    // replace more/stop with comment/submit
    remove(more);
    more.setVisible(false);
    remove(stop);
    stop.setVisible(false);
    add(comment);
    comment.setVisible(true);
    add(submit);
    submit.setVisible(true);
}
```

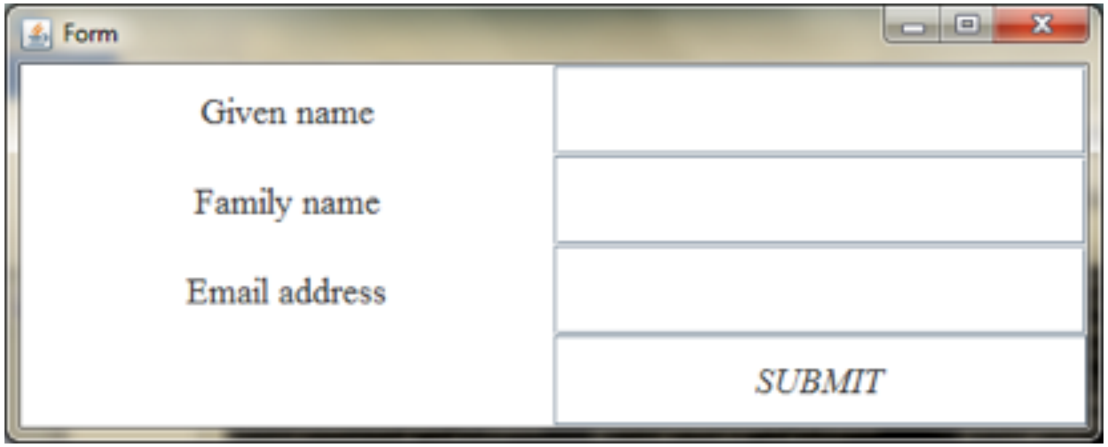
Example: form

```
// will be called when stop button is pressed
void doStop()
{   file.close();
    System.exit(0);
}
```

Example: form

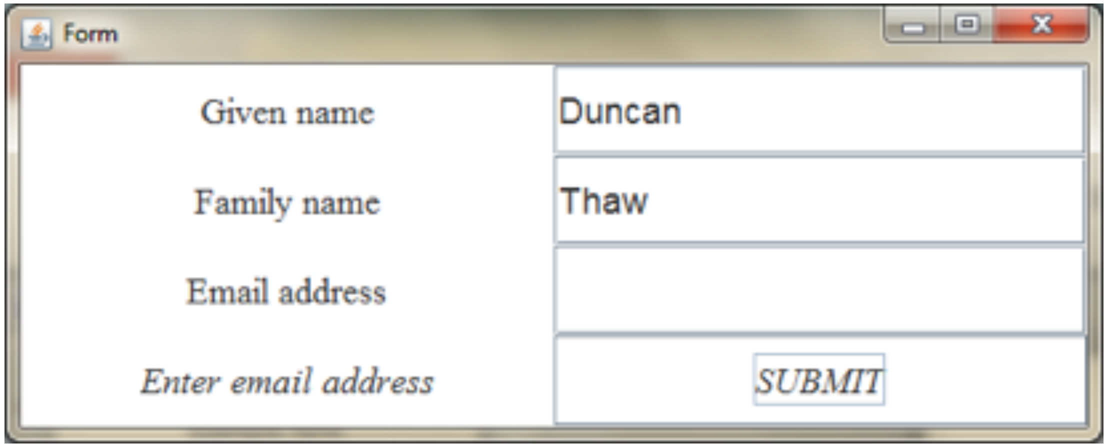
```
   // setup the printwriter to write to a specified file
   public void setup() throws IOException
   {  file = new PrintWriter
                  (new FileWriter("register.log"),true);
   }
}

class TestForm
{  public static void main(String [] args)
    throws IOException
   {  Form f;
      f = new Form(); // create form
      ...
      f.setup(); // and setup file output
   }
}
```

Example: form



Form

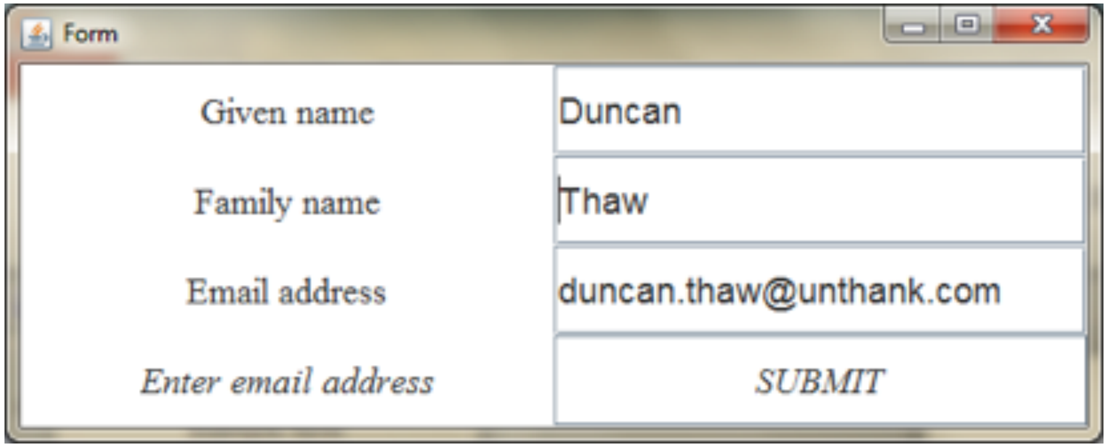
Given name	
Family name	
Email address	
	SUBMIT



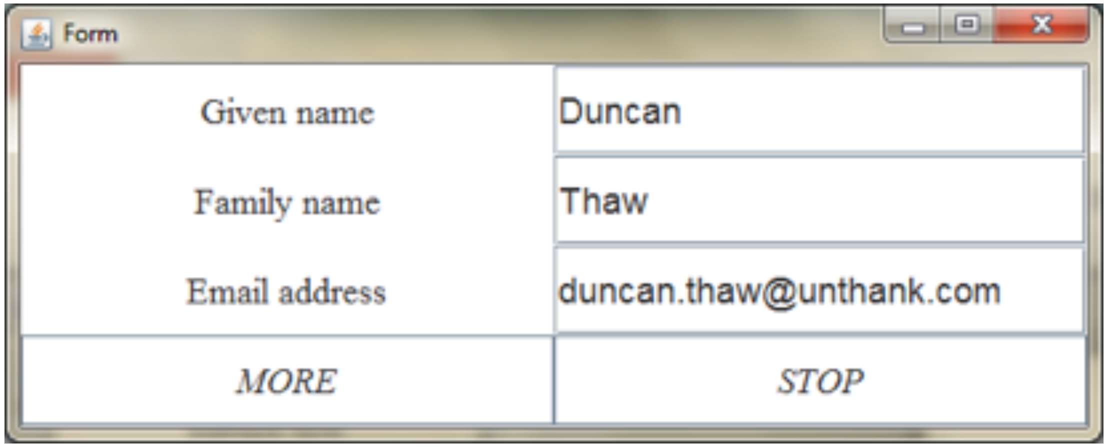
Form

Given name	Duncan
Family name	Thaw
Email address	
Enter email address	*SUBMIT*

Example: form



Given name	Duncan
Family name	Thaw
Email address	duncan.thaw@unthank.com
Enter email address	*SUBMIT*



Given name	Duncan
Family name	Thaw
Email address	duncan.thaw@unthank.com
MORE	*STOP*

THAT’S IT

Next Lecture

- Lots more examples