บทที่ 4

การพัฒนาระบบ

โปรแกรม FOOD นี้พัฒนาขึ้นด้วยโปรแกรมภาษาจาวาบนระบบปฏิบัติการวินโดว์ โดยใช้องค์ประกอบจากไลบรารีของจาวาดีเวลลอปเมนท์คิด เวอร์ชัน 1.2 (Java Development Kit : JDK1.2) การสร้างส่วนติดต่อกับผู้ใช้ จะเรียกใช้คลาสต่างๆ ในแพ็กเกจสวิง

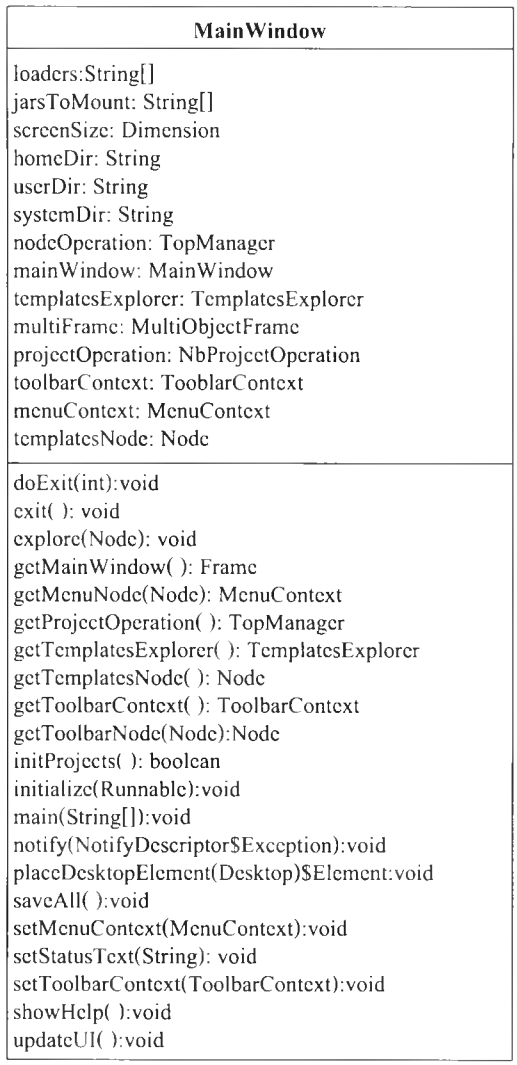
ในบทนี้เป็นการอธิบายรายละเอียดการทำงานของคลาสต่างๆ ภายใน 4 แพ็กเกจ คือ แพ็กเกจสำหรับสร้างโปรแกรมประยุกต์ แพ็กเกจสำหรับจัดการจาวาบีน แพ็กเกจสำหรับจัดการรูปแบบการออกแบบ และแพ็กเกจสำหรับสร้างชุดคำสั่ง

4.1 การสร้างแพ็กเกจสำหรับสร้างโปรแกรมประยุกต์

ในแพ็กเกจเมนวินโดว์จะมีคลาสชื่อเมนวินโดว์เป็นคลาสหลักในการสร้างเมนูให้ผู้ใช้เลือกการทำงาน โดยมีคลาสทอปเมเนเจอร์ ทำหน้าที่จัดการเกี่ยวกับการแสดงผลบนหน้าจอ คลาสเอ็กโพเลอร์เมเนเจอร์ ทำหน้าที่แสดงรายการออปเจคของงานที่กำลังทำอยู่ คลาสฟอร์มอีดิเตอร์ทำหน้าที่แสดงผลชุดคำสั่งที่สร้างอัตโนมัติให้ผู้ใช้ทำการเพิ่มชุดคำสั่ง คลาสแอพเพล็ตเมนเป็นคลาสที่สร้างส่วนติดต่อกับผู้ใช้เพื่อจัดวางองค์ประกอบ และคลาสคอมไพล์เลอร์เอ็กซีคิวเตอร์ทำหน้าที่เรียกใช้จาวาคอมไพล์เลอร์ในการแปลชุดคำสั่งเป็นคลาสไฟล์ และใช้รันโปรแกรมประยุกต์ที่สร้าง จากรูปที่ 3.5 มีรายละเอียดภายในแพ็กเกจดังนี้

4.1.1 คลาสวินโดว์หลัก

คลาสเมนวินโดว์ เป็นคลาสหลักในการสร้างเมนูเพื่อติดต่อกับผู้ใช้ ในการสร้างโปรแกรมประยุกต์ สร้างจาวาบีน เรียกใช้รูปแบบการออกแบบ คอมไพล์โปรแกรม เป็นต้น โดยจะทำหน้าที่เรียกใช้คลาสที่สัมพันธ์กับเมนูที่เลือก ภายในคลาสจะมีเมทธอดเพื่อทำหน้าที่ต่างๆได้แก่ กำหนดติดตั้งค่าเริ่มต้นให้กับระบบ (initialize) กำหนดการทำงานเริ่มต้นเมื่อเริ่มสร้างโปรแกรมประยุกต์ (initProjects) กำหนดติดตั้งทูลบาร์ (setToolbar) สร้างเมนู (setMenuContext) เป็นต้น รายละเอียดภายในคลาสเมนวินโดว์ ดังแสดงในรูปที่ 4.1 และรายละเอียดเมทธอดในคลาสเมนวินโดว์ดังแสดงในรูปที่ 4.2



รูปที่ 4.1 รายละเอียดภายในคลาสเมนวินโดว์

```
public static void main(String args[])
    throws SecurityException
  {
    if(mainWindow != null)
      throw new SecurityException();
    System.getProperties().put("netbeans.design.time", "true");
    homeDir = System.getProperty("netbeans.home");
    if(homeDir == null)
      doExit(1);
    if(System.getProperty("netbeans.user") == null)
      System.getProperties().put("netbeans.user", homeDir);
    userDir = System.getProperty("netbeans.user");
    File file = new File(homeDir);
    File file1 = new File(userDir);
    if(!file.exists())
    {
      System.out.println(mainBundle.getString("CTL_Netbeanshome_notexists"));
      doExit(2);
    }
    if(!file.isDirectory())
    {
      System.out.println(mainBundle.getString("CTL_Netbeanshome1"));
      doExit(3);
    }
    if(!file1.exists())
    {
      System.out.println(mainBundle.getString("CTL_Netbeanshome2"));
      doExit(4);
    }
    if(!file1.isDirectory())
    {
      System.out.println(mainBundle.getString("CTL_Netbeanshome3"));
      doExit(5);
    }
    systemDir = userDir + File.separator + "system";
    File file2 = new File(systemDir);
    if(file2.exists())
    {
      if(!file2.isDirectory())
      {
        Object aobj[] = {
          userDir
        };
        System.out.println((new
MessageFormat(mainBundle.getString("CTL_CannotCreate_text"))).format(((Object) (aobj))));
        doExit(6);
      }
    }
    else
    if(!file2.mkdir())
    {
      Object aobj1[] = {
        userDir
      };
      System.out.println((new
MessageFormat(mainBundle.getString("CTL_CannotCreateSysDir_text"))).format(((Object) (aobj1))));
      doExit(7);
    }
    String s = System.getProperty("os.name");
    if(s.startsWith("Windows"))
      try
      {
        //UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
      UIManager.setLookAndFeel("com.sun.java.swing.plaf.metal.MetalLookAndFeel");

      }
      catch(Exception _ex) { }
    ToolTipManager.sharedInstance().setLightWeightPopupEnabled(false);
    parseCommandLine(args);
    topBundle = NetbeansBundle.getBundle("com.netbeans.developer.locales.TopBundle");
}
```

รูปที่ 4.2 รายละเอียดเมทธอดเมนในคลาสเมนวินโดว์

4.1.2 คลาสจัดการหลัก

คลาสทอปเมเนเจอร์ เป็นคลาสหลักในการเตรียมสภาพการทำงานเมื่อผู้ใช้เลือกเมนูสร้างโปรแกรมประยุกต์ โดยจะทำหน้าที่เรียกคลาสที่เกี่ยวข้องอีก 3 คลาส คือ คลาสกำหนดสภาพแวดล้อมการทำงาน(IDESetings) คลาสกำหนดค่าโปรเจคใหม่(ProjectOperation) และคลาสรายการต้นแบบงาน(TemplateExplorer) ภายในคลาสทอปเมเนเจอร์จะมีเมทธอดเพื่อทำหน้าที่ต่างๆได้แก่ กำหนดติดตั้งค่าเริ่มต้นให้กับระบบ(initialize) กำหนดการทำงานเริ่มต้นเมื่อเริ่มสร้างโปรแกรมประยุกต์(initProjects) กำหนดติดตั้งทูลบาร์(setToolbar) สร้างเมนู(setMenuContext) เป็นต้น รายละเอียดภายในคลาสทอปเมเนเจอร์ ดังแสดงในรูปที่ 4.3 รายละเอียดภายในคลาสไอดีอีเซ็ทติ้ง ดังแสดงในรูปที่ 4.4 รายละเอียดภายในคลาสโปรเจคโอเปอเรชัน ดังแสดงในรูปที่ 4.5 และรายละเอียดภายในคลาสเทมเพลสเอ็กโพเลอร์ ดังแสดงในรูปที่ 4.6

TopManager
properIcon: Image Prop_Display_name: String ret:Node[] map: hashtable staticActions : systemAction[]
added(SyatemOption):void getActions():systemAction[] getCurrentName():String initialize():void removed(SystemOption):void rename(String):void thisNodeChange():void

รูปที่ 4.3 รายละเอียดภายในคลาสทอปเมเนเจอร์

IDESetings
PROP_OUTPUT_LEVEL: String PROP_LOOK_AND_FEEL: String OUTPUT_MINIMUM: int OUTPUT_NORMAL: int OUTPUT_MAXIMUM: int
displayName(): String getBeanInfoSearchPath():String[] getLoadedBeans(): Hashtable getPropertyEditorSearchPath(): String[] setBeanInfoSearchPath(String[]): void setLoadedBeans(Hashtable): void setLookAndFeel(LookAndFeel): void setOutputLevel(int):void setPropertyEditorSearchPath(String[]): void

รูปที่ 4.4 รายละเอียดภายในคลาสไอดีอีเซ็ทติ้ง

ProjectOperation
DEFAULTNAME: String currentProject: DataProject
exit(): void getCurrentProject():DataProject int(): boolean opern(DataProject, InputStream): void save(ObjectOutputStream): void saveCurrentProject(): void

รูปที่ 4.5 รายละเอียดภายในคลาสโปรเจคโอเปอเรชัน

TemplatesExplorer
okButton: ButtonBarButton cancelButton: ButtonBarButton explorerManager: ExplorerManager selectedTemplate: DataObject tabs: JTabbedPane currentPanel: JPanel previousRootNode: Node nodeToTab: Hashtable tabToNode: Hashtable internalChange: boolean listener: NodeListener nameListener: NodeListener
buttonPressed(ButtonBar$ButtonBarEvent):void getPreferredSize( ):Dimension updateCurrentTab( ):void updateTabs( ):void writeReplace( ):Object

รูปที่ 4.6 รายละเอียดภายในคลาสเทมเพลสเอ็กโพเลอร์

4.1.3 คลาสสำรวจออปเจคในโปรเจค

คลาสเอ็กโพเลอร์เมเนเจอร์เป็นคลาสแสดงรายการออปเจคทั้งหมดที่ได้สร้างขึ้นในระหว่างการสร้างโปรแกรมประยุกต์โดยแสดงเป็นโครงสร้างต้นไม้ ผู้ใช้สามารถคลิกเม้าส์ที่ออปเจคเพื่อแสดงวินโดว์กำหนดคุณสมบัติ ภายในคลาสเอ็กโพเลอร์เมเนเจอร์จะมีเมทธอดเพื่อทำหน้าที่ต่างๆได้แก่ อ่านรายการออปเจคทั้งหมด(getExploreredContext) เลือกออปเจค(getSelectedNodes) กำหนดติดตั้งแสดงโครงสร้างต้นไม้(setExplorerContext) เป็นต้น รายละเอียดภายในคลาสเอ็กโพเลอร์เมเนเจอร์ ดังแสดงในรูปที่ 4.7 รายละเอียดเมทธอดเซ็ทเอ็กโพเลอร์คอนเท็กซ์ภายในคลาสเอ็กโพเลอร์เมเนเจอร์ ดังแสดงในรูปที่ 4.8

Explorer Manager
selected Nodes Listener:Node Listener selected Nodes Parents:Hashtable root context Listener:Node Listener root Context Path:Vector explored Context Listener:Node Listener explored Context Path:Vector vetoable Support:Vetoable Change Support property Support:Property Change Support root Context:Node explored Context:Node selected Nodes:Node []
addPropertyChangeListener(PropertyChangeListener):void addVetoableChangeListener(VetoableChangeListener):void getExploredContext( ):Node getPathTo(Node):Vector getRootContext( ):Node getSelectedNodes( ):Node[] init ():void is Under Root (Node):boolean read Object (Object Input Stream):void remove Listeners ():void remove Property Change Listener (Property Change Listener):void remove Vetoable Change Listener (Vetoable Change Listener):void set Explored Context (Node):void set Root Context (Node):void set Selected Nodes (Node []):void write Object (Object Output Stream):void

รูปที่ 4.7 รายละเอียดภายในคลาสเอ็กโพเลอร์เมเนเจอร์

```
public final void setExploredContext(Node node){
    if (exploredContext != null && exploredContext.equals(node))
        return;
    if (!isUnderRoot(node))
        return;
    try
    {
        setSelectedNides(new Node[0]);
    }
    catch(PropertyVetoException _ex)
    {
        throw new InternalError(explorerBundle.getString("Err_MustNotVetoEmptySelection"));
    }
    Node nodel = exploredContext;
    exploredContext = node;
    Vector   vector = getPathTo(exploerdContext, rootContext);
    for (int i = k; i >=0; i--)
        if (flag && exploredContextPath.size() > 0 && exploredContextPath.lastElement() ==
                vector.elementAt(i))
        {
          exploredContextPath.removeElementAt(exploredContextPath.size() –1;
        }
        else
        {
          ((Node)vector.elementAt(i)).addNodeListener(exploredContextListener);
          flag = false;
        }
    k = exploredContextPath.size();
    for (int j = 0; j < k; j++)
        ((Node)exploredContextPath.elementAt(j)).removeListener(exploredContextListener);
    exploredContextPath = vector;
    if (propertySupport != null)
        propertySupport.firePropertyChange("exploredContext", model, exploredContext);
}
```

รูปที่ 4.8 รายละเอียดเมทธอดเซ็ทเอ็กโพเลอร์คอนเท็กซ์ภายในคลาสเอ็กโพเลอร์เมเนเจอร์

4.1.4 คลาสแสดงผลชุดคำสั่งที่สร้าง

คลาสฟอร์มอีดิเตอร์เป็นคลาสแสดงผลโครงชุดคำสั่งที่ระบบสร้างขึ้นให้อัตโนมัติในระหว่างการสร้างโปรแกรมประยุกต์โดยแสดงเป็นวินโดว์ให้ผู้ใช้สามารถแก้ไข เพิ่มเติมชุดคำสั่ง ภายในคลาสฟอร์มอีดิเตอร์จะมีเมทธอดเพื่อทำหน้าที่ต่างๆได้แก่ แสดงโครงชุดสำสั่งของออปเจคที่ถูกเลือก (getComponentInspector) แสดงโครงชุดคำสั่งเริ่มต้น(getJavaSettings) เป็นต้น รายละเอียดภายในคลาสฟอร์มอีดิเตอร์ ดังแสดงในรูปที่ 4.9

FormEditor
formSettings:FormLoaderSettings javaSettings:JavaLoaderSettings designModeAction:DesignModeAction paletteAction:PaletteAction componentInspector:FormEditorSComponentInspector explorerManager:ExplorerManager
defaultComponentInit(RADVisualNode):void default MenuInit(RADMenuItemNode):void formActivated(FormManager):void getComponentInspector( ):FormEditorSComponentInspector getExplorerManager( ):ExplorerManager getFormSetting( ):FormLoaderSetting getJavaSetting( ):JavaLoaderSettings PaletteAction getPaletteAction( )

รูปที่ 4.9 รายละเอียดภายในคลาสฟอร์มอีดิเตอร์

4.1.5 คลาสแอพเพล็ตเมน

คลาสแอพเพล็ตเมนเป็นคลาสที่สร้างฟอร์มวินโดว์สำหรับจัดวางองค์ประกอบในการสร้างโปรแกรมประยุกต์ มีการเรียกใช้คลาสที่เกี่ยวข้องคือ คลาสแอพเพล็ตซับพอร์ต และคลาสแอพเพล็ตเซ็ตติ้ง ภายในคลาสแอพเพล็ตเมนจะมีเมทธอดเพื่อทำหน้าที่ต่างๆได้แก่ สร้างแอพเพล็ตวิวเวอร์เพื่อใช้ในการจัดวางองค์ประกอบ(processAppleteViewer) เป็นต้น รายละเอียดภายในคลาสแอพเพล็ตเมน ดังแสดงในรูปที่ 4.10

AppletMain
appletViewerName:String appletPanel:String options:AppletSettings
main(String[]):void processAppletviewer(String):void processExternal(String):void

รูปที่ 4.10 รายละเอียดภายในคลาสแอพเพล็ตเมน

4.1.6 คลาสคอมไพเลอร์เอ็กซีคิวเตอร์

คลาสคอมไพล์เลอร์เอ็กซีคิวเตอร์เป็นคลาสที่ทำหน้าที่เรียกใช้จาวาคอมไพล์เลอร์ในการแปลชุดคำสั่งเป็นคลาสไฟล์ และใช้รันโปรแกรมประยุกต์ที่สร้าง มีการเรียกใช้คลาสที่เกี่ยวข้องคือ คลาสคอมไพเลอร์เซ็ทติ้ง และคลาสคอมไพเลอร์ซิสโปรเซส ภายในคลาสคอมไพล์เลอร์เอ็กซีคิวเตอร์จะมีเมทธอดเพื่อทำหน้าที่ต่างๆได้แก่ กำหนดตำแหน่งเส้นทางการจัดเก็บคลาสไฟล์(addClassPath) กำหนดค่าตัวแปรต่างๆในการคอมไพล์(constructArgument) เป็นต้น รายละเอียดภายในคลาสคอมไพล์เลอร์เอ็กซีคิวเตอร์ ดังแสดงในรูปที่ 4.11

CompilerExecutor
settings:ExternalCompilerSetting path:String classPath:String file:String eCompiler:externalCompiler holder:Object mewProc:CompilerSysProcess process:Process
addClassPath(String):void constructArgument(String):String displayName( ):String getClasspath(String[]):String

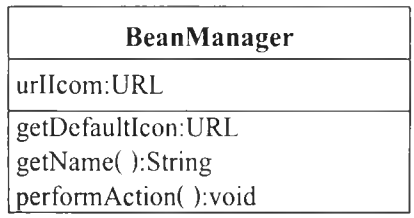
รูปที่ 4.11 รายละเอียดภายในคลาสคอมไพล์เลอร์เอ็กซีคิวเตอร์

4.2 การสร้างแพ็กเกจสำหรับจัดการจาวาบีน

ในแพ็กเกจบีนเมเนเจอร์ จะมีคลาสบีนเมเนเจอร์ เป็นคลาสหลักในการเรียกใช้คลาสอื่นๆ ในขั้นตอนการสร้างหรือนำเข้าจาวาบีน คลาสบีนเมเนเจอร์ไดอะล็อก มีหน้าที่จัดการกรอบข้อความเพื่อรับค่าจากผู้ใช้ และ คลาสบีนเจนเนอร์เรเตอร์ มีหน้าที่สร้างชุดคำสั่งจาวาบีน ความสัมพันธ์ของคลาสต่างๆในแพ็กเกจสำหรับจัดการจาวาบีน จากรูปที่ 3.6 มีรายละเอียดในแพ็กเกจดังนี้

4.2.1 คลาสจัดการจาวาบีน

คลาสบีนเมเนเจอร์ เป็นคลาสหลักของส่วนจัดการจาวาบีน โดยจะรับคำสั่งจากเมนู เมื่อผู้ใช้ต้องการสร้างจาวาบีนจะทำการเรียกคลาสบีนเมเนเจอร์ไดอะล็อก เพื่อรับข้อมูลจากผู้ใช้ รายละเอียดภายในคลาสบีนเมเนเจอร์ ดังแสดงในรูปที่ 4.12 และรายละเอียดเมทธอดในคลาสบีนเมเนเจอร์ ดังแสดงในรูปที่ 4.13



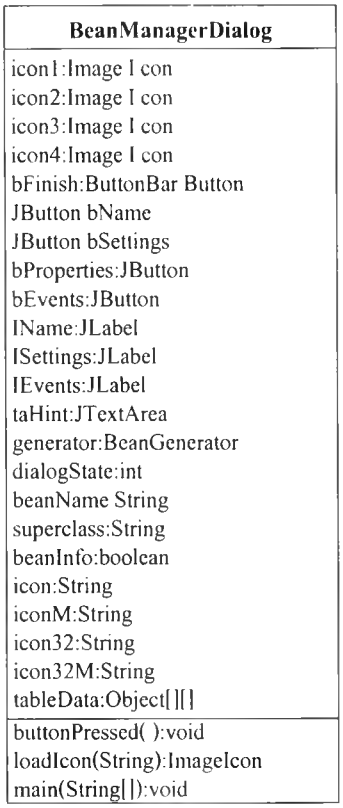
รูปที่ 4.12 รายละเอียดภายในคลาสบีนเมเนเจอร์

```
public class BeanManager extends CallableSystemAction
{

   public BeanManager()
   {
   }

   public URL getDefaultIcon()
   {
      if(urlIcon == null)
         urlIcon = getClass().getResource("beanManager.gif");
      return urlIcon;
   }


   public String getName()
   {
      return BeanWizardDialog.bundle.getString("LAB_BeanWizardAction");
   }

   public void performAction()
   {
      new BeanManager();
   }
}
```

รูปที่ 4.13 รายละเอียดเมทธอดในคลาสบีนเมเนเจอร์

4.2.2 คลาสจัดการกรอบข้อความจาวาบีน

คลาสบีนเมเนเจอร์ไดอะล็อก เป็นคลาสที่จะทำหน้าที่ติดต่อกับผู้ใช้ในการรับข้อมูล ชื่อของจาวาบีน ไฟล์รูปภาพ ชื่อตัวแปร ชื่อเมทธอด เพื่อนำมาสร้างจาวาบีน โดยจะแบ่งเป็น 3 ขั้นตอน คือ กำหนดชื่อจาวาบีน กำหนดซุปเปอร์คลาส กำหนดไอคอน และกำหนดตัวแปรคุณสมบัติให้แก่จาวาบีน รายละเอียดภายในคลาสบีนเมเนเจอร์ไดอะล็อก ดังแสดงในรูปที่ 4.14 และรายละเอียดเมทธอด พรอบเพอร์ตี้ไดอะล็อกในคลาสบีนเมเนเจอร์ไดอะล็อก ดังแสดงในรูปที่ 4.15



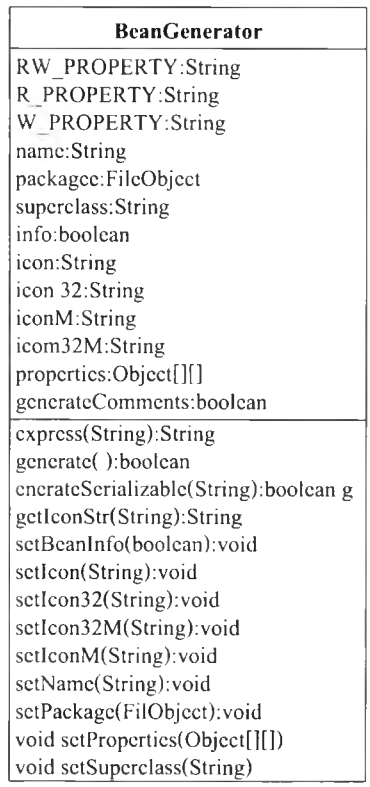
รูปที่ 4.14 รายละเอียดภายในคลาสบีนเมเนเจอร์ไดอะล็อก

```
PropertiesDialog(){
    super(null, new ButtonBar(new ButtonBarButon[]{
        new ButtonBarButton (getString("Button_Ok")), new ButtonBarButton(getString
        ("Button_cancel")) }), true);
    newProperty = (new Object[]{
        getString("Label_new_property"), "String", BeanGenerator.RW_PROPERTY,
        new Boolean(true), new Boolean(false), new Boolean(true)});
    setTitle(getString("Dialog_properties_title"));
    Jpanel jpanel = new Jpanel();
    jpanel.setLayout(new BorderLayout(10, 10));
    jpanel.setBorder(new EmptyBorder(10, 10, 10));
    jpanel jpanell = new Jpanel( );
    jpanell.setBorder(new CompoundBorder(new EtchedBorder(), new EmptyBorder(5, 5, 5, 5)));
    jpanell.setLayout(new BorderLayout());
    JtextArea jtextarea = new JtextArea();
    jtextarea.setLineWrap(true);
    jtextarea.setEditable(false);
    jtextarea.setenabled(false);
    jtextarea.setWrapStyleWord(true);
    jtextarea.setDisabledTextcolor(UIManager.getColor("Label.foreground"));
    jtextarea.setBackground(UIManager.getColor("Label background"));
    jtextarea.setText( getString("Kint_properties"));
    jpanell.add(jtextarea, "Center");
    jpanel.add("North", jpanell);
    if (tableData !=null){
        int j = tableData.length;
        data = new Object[tableData.length][];
        for(int i= 0; i , j; i++){
            data[i] = new Object[tableData[i].length];
            System.arraycopy(((Object) (tableData[i])), 0, ((Object) (data[i])), 0, tableData[i].length);{
    }else{
        data = new Object[0][];}
```

รูปที่ 4.15 รายละเอียดเมทธอดพรอบเพอร์ตี้ไดอะล็อกในคลาสบีนเมเนเจอร์ไดอะล็อก

4.2.3 คลาสสร้างชุดคำสั่งจาวาบีน

คลาสบีนเจนเนอร์เรเตอร์ เป็นคลาสที่จะทำหน้าที่รับข้อมูลจากคลาสบีนเมเนเจอร์ ไดอะล็อก เพื่อนำมาสร้างชุดคำสั่งจาวาบีน รายละเอียดภายในคลาสบีนเจนเนอร์เรเตอร์ดังแสดงใน รูปที่ 4.16 และรายละเอียดเมทธอด เจนเนอร์เรทในคลาส บีนเจนเนอร์เรเตอร์ ดังแสดงในรูปที่ 4.17



รูปที่ 4.16 รายละเอียดภายในคลาสบีนเจนเนอร์เรเตอร์

```
boolean generate( ){
    FileObject fileobject = packagee;
    String s = fileobject.getPackageName(‘  ‘);
    fileLock filelock = null;
    try{
        FileObject fileobject1 = fileobject.createData(name, “Java”);
        filelock = fileobject1.lock( );
        Object obj = new BufferedOutputStrearm(filcobject1.getOutputStream(filelock));
        dataoutputstream = new DataOutputStream(((jave.io.OutputStream) (obj)));
        dataoutputstream.writeBytes(s.length( ) ,= 0 ? “ “ : “\npackage “ + s + “;\n\n\n”);
        if (generateComments)
            dataoutputstream.writeBytes(“/**\n * A “ + name + “JavaBean.\n” + “ *\n” +
            “ * @version 1.00\n” + “ */ \n”);
        dataoutputstream.writeBytes(public class “ + name + “extends “ + superclass +
        “\n” + (generateSerializable (superclass) ? “implements java.io.Serializable “ : “”) + “(\n\n”);
        int 1 = properties.length;
        boolean flag4 = false;
        boolean flag5 = false;
        int i’
        for (i = 0 ; i < 1; i++){
            Object aobj[ ] = properties[i];
            if(((Boolean)aobj[3]).booleanValue())
                flag4 = true;
            if(((Boolean)aobj[4]).booleanValue())
                flag5 = true;
```

รูปที่ 4.17 รายละเอียดเมทธอดเจนเนอร์เรทในคลาสบีนเจนเนอร์เรเตอร์

```
    if(((Boolean)aobj[5]).booleanValue()){
        String sl = (String)aobj [0];
        sl = sl.substring(0, 1).toLowerCase() + sl.substring(1)
        if(generateComments)
            dataoutputstream.writeBytes(" /**\"" + sl + "\" property value *\n");
        dataoutputstream.writeBytes(" private " + aobj[1] + " " + sl + ",\n); }
}
if (i > 1)
    dataoutputstream.writeBytes9"\n");
if (flag4){
    if (generateComments)
        dataoutputstream.writeBytes(" /** The support for property changes *\n);
    dataoutputstream.writeBytes("private java.beans.VetoableChangeSupport vetoableChangeSupport; \n);}
if (flag5){
    if (generateComments)
        dataoutputstream.writeBytes(" /** The support for property changes *\n);
    dataoutputstream.writeBytes("private java.beans.VetoableChangeSupport vetoableChangeSupport; \n);}
if (generateComments)
    dataoutputstream.writeBytes(" /**\n * Constructs a new " + name+ "JavaBean \n" + : *\n");
dataoutputstream.writeBytes("public "+name+ "() [\n");
if(flag4)
    dataoutputstream.writeBytes("propertyChangeSupport=new java.beans.PropertyChangeSupport (this):\n);
if(flag5)
    dataoutputstream.writeBytes("propertyChangeSupport=new java.beans.PropertyChangeSupport (this):\n);
dataoutputstream.writeBytes(" }\n\n");
for (int j = 0 ; , 1; j++) {
    Object aobj1[ ] = properties[j];
    String s2 = ((String_aobj1[0]).trim();
    S2 = s2.substring(0, 1).toUpperCase() + s2.substring(1);
    String s3 = s2.substring(0,, 1).toLowerCase()+s2.substring(1);
    if(aobj1[2].equals(RW_PROPERTY) || aobj1[2].equals)R_PROPERTY))}
        if(generatComments)
            dataoutputstream.writeBytes(" ///**\n Getter method for the "" +s3 + "\" property\n"+
            " *\n"+" **@return "+" The current value of this property \n:+" *\n");
        dataoutputstream.writeBytes(" return" + s4 + "get" +s2+" () {\n");
        if(((boolean)aobj1[5].booleanValue())
            dataoutputstream.writeBytes(" return "+s3+";\n");
        else
            dataoutputstream.writeBytes(" //PENDING Write a code for getting the property value \n");
        dataoutputstream.writeBytes(; }\n\n"); }
    if(generateComments)
        dataoutputstream.writeBytes(: /**\n *Setter method ofr the \""+s3+ "\" property \n"+
        " *\n+" * @param value The new value of this property. \n" +" *\n");
    dataoutputstream.writeBytes(" public void set" +s2+ "("+s4+value)");
    if(((boolean)aobj1[4]). booleanValue())
        dataoutputstream.writeBytes("throws java.beans.PropertyWetoException {\n");
    else
        dataoutputstream.writeBytes("{\n");
    String s6 = (String)simple.get(s4X;
    String s8;
    if (s6 = = null) {
        s6 = "value";
        s8 = s3; }
    else {
        s8 = "new" + s6 + "("+s3")";
        s6 = "new" + s6 + "(value); }
    if(((Boolean_aobj1[5].booleanValue()) {
        if(((Boolean)aobj1[4].booleanValue())
            dataoutputstream.writeBytes("vetoableChangeSupport fireVetoableChange (\"" + s3 + "\", "+s8 +,
            "+ s6 +");\n);
        if(((Boolean)aobj1[3].booleanValue())
            dataoutputstream.writeBytes("Object oldValue = "+s8+ ";\n");
        dataoutputstream.writeBytes(" "+s3+ " = value;\n;)
        if(((Boolean)aobj1[3].booleanValue())
            dataoutputstream.writeBytes(" propertyChangeSupport firePropertyChange (\"" +s3+"\".oldValue.'
    }
```

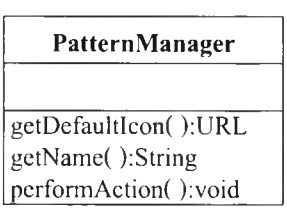
รูปที่ 4.17 รายละเอียดเมทธอดเจนเนอร์เรทในคลาสบีนเจนเนอร์เรเตอร์ (ต่อ)

4.3 การสร้างแพ็กเกจสำหรับจัดการรูปแบบการออกแบบ

ในแพ็กเกจสำหรับจัดการรูปแบบการออกแบบจะมีคลาสจัดการรูปแบบการออกแบบเป็นคลาสหลักในการเรียกใช้คลาสอื่นๆในการสร้างโครงชุดคำสั่งของรูปแบบการออกแบบ คลาสจัดการกรอบข้อความเป็นส่วนรับค่าจากผู้ใช้ และคลาสสร้างโครงชุดคำสั่งรูปแบบการออกแบบจะสร้างโครงชุดคำสั่ง จากรูปที่ 3.7 มีรายละเอียดในแพ็กเกจดังนี้

4.3.1 คลาสจัดการรูปแบบการออกแบบ

คลาสแพทเทิร์นเมเนเจอร์เป็นคลาสหลักของส่วนจัดการรูปแบบการออกแบบ โดยจะรับคำสั่งจากเมนูเมื่อผู้ใช้ต้องการสร้างโครงชุดคำสั่งของรูปแบบการออกแบบจะทำการเรียกคลาสแพทเทิร์นเมเนเจอร์ไดอะล็อกเพื่อรับข้อมูลจากผู้ใช้ รายละเอียดภายในคลาสแพทเทิร์นเมเนเจอร์ดังแสดงในรูปที่ 4.18 และรายละเอียดเมทธอดในคลาส แพทเทิร์นเมเนเจอร์ดังแสดงในรูปที่ 4.19



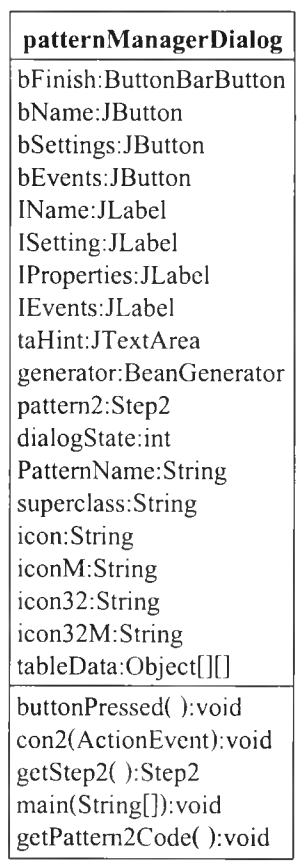
รูปที่ 4.18 รายละเอียดภายในคลาสแพทเทิร์นเมเนเจอร์

```
public class PatternManager extends CallableSystemAction
{
    public PatternManager()
    {
    }
    public URL getDefaultIcon()
    {
        if (urlIcon == null)
            urlIcon = getClass().getResource("patternManager.gif");
        return urlIcon;
    }
    public HelpCtx getHelpCtx()
    {
        return new HelpCtx("usergd-action", "USERGD-ACTION-TABLE-1");
    }
    public String getName()
    {
        return PatternManagerDialog.bundle.getString("LAB_PatternWizardAction");
    }
    public void performAction()
    {
        new PatternManagerDialog();
    }
}
```

รูปที่ 4.19 รายละเอียดเมทธอดในคลาสแพทเทิร์นเมเนเจอร์

4.3.2 คลาสจัดการกรอบข้อความรูปแบบการออกแบบ

คลาสแพทเทิร์นเมเนเจอร์ไดอะล็อกเป็นคลาสที่จะทำหน้าที่ติดต่อกับผู้ใช้ในการรับข้อมูลการเลือกรูปแบบการออกแบบ ชื่อไฟล์ของรูปแบบการออกแบบ ชื่อตัวแปร ชื่อเมทธอด เพื่อนำมา สร้างโครงชุดคำสั่งรูปแบบการออกแบบ รายละเอียดภายใน แพทเทิร์นเมเนเจอร์ไดอะล็อก ดังแสดงในรูปที่ 4.20 และรายละเอียดเมทธอดในคลาส แพทเทิร์นเมเนเจอร์ไดอะล็อก ดังแสดงในรูปที่ 4.21



รูปที่ 4.20 รายละเอียดภายในคลาสแพทเทิร์นเมเนเจอร์ไดอะล็อก

```
public PatternManagerDialog(Frameframe)
{
        super(frame, true);
        bFinish = new ButtonBarButton(getString("Button_finish));
        dialogState = 0;
        PatternName =  "newPattern";
        superclass =  "java.awt.Component";
        beanInfo = true;
        icon = bundle.getString("Label_none");
        iconM = bundle.getString("Label_none");
        icon32 = bundle.getString("Label_none");
        icon32M = bundle.getString("Label_none");
        getButtonBar().setButtons(new ButtonBarButton[0], new ButtonBarButton[]{
            bFinish, new ButtonBarButton(bundle.getString("Button_cancel"))
});
```

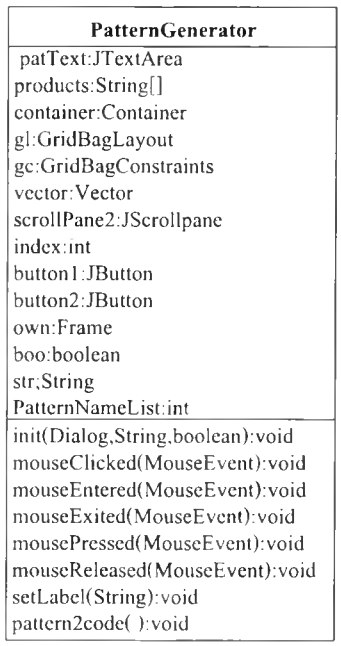
รูปที่ 4.21 รายละเอียดเมทธอดในคลาสแพทเทิร์นเมเนเจอร์ไดอะล็อก

```
generator = new PatternGenertor();
bFinish.setEnabled(false);
JPanel jpanel – new JPanel (new borderLayout910, 10));
jpanel.setborder (new Emptyborder(10, 10, 10));
JPanel1 jpanell = new JPanel();
jpanel1.setBorder(new CompoundBorder(new Etchedborder(), new EmptyBorder (5, 5, 5, 5)));
jpanel1.setLayout(new borderLayout());
jpanel1.add(taHint, "Center");
jpanel1.add("North", jpanel1);
jpanel1 = new JPanel ();
GridBagLayout gridbaglayout = new GridBagLayout ();
GridBagConstraints gridbagconstraints = new GridBagConstraints();
jpanel1.setLayout (gridbaglayout);
jpanel1.setborder(new Emptyborder (10, 10, 10, 10));
Jabel jlabel = new Jlabel (icon1);
gridbagconstraints.gridwidth = 1;
gridbagconstraints.insets = new Insets (0, 0, 0, 10));
gridbaglayout.setConstraints(jlabe, gridbagconstraints);
jpanel1.add(jlabel);
bName = new Jbutton(bundle.getString("Button_name"));
    public void actionPerformed (ActionEvent actionevent)
    {
        new NameDialog();
    }
});
```

รูปที่ 4.21 รายละเอียดเมทธอดในคลาสแพทเทิร์นเมเนเจอร์ไดอะล็อก(ต่อ)

4.3.3 คลาสสร้างโครงชุดคำสั่งของรูปแบบการออกแบบ

คลาสแพทเทิร์นเจนเนอร์เรเตอร์ เป็นคลาสที่จะทำหน้าที่รับข้อมูลจากผู้ใช้ในการเลือกรูปแบบการออกแบบ การกำหนดชื่อไฟล์ของรูปแบบการออกแบบ ชื่อตัวแปร ชื่อเมทธอด เพื่อนำมาสร้างโครงชุดคำสั่งรูปแบบการออกแบบ รายละเอียดภายในคลาส แพทเทิร์นเจนเนอร์เรเตอร์ ดังแสดงในรูปที่ 4.22 และรายละเอียดเมทธอดแพทเทิร์นทูโค้ด(pattern2code)ในคลาสแพทเทิร์นเจนเนอร์เรเตอร์ ดังแสดงในรูปที่ 4.23



รูปที่ 4.22 รายละเอียดภายในคลาสแพทเทิร์นเจนเนอร์เรเตอร์

```
public void Pattern2code( )
{
        int n = 0;
        char c [] = new char [128];
    static String replace(String str,String pattern, String replace){
        int s = 0;
        int e = 0;
        StringBuffer result = new StringBuffer( );
        While ((e=str.indexOf (pattern,s))>=0){
            result.append(str.substring(s,e));
            result.append(replace);
            s-e+pattern.length();
        }
        result.append(str.substring(s));
        return result.toString();
    }
    public void tocode(String pattern, String before, String after, String patfile)
    {
        String t_pattern = pattern+ "jave";
        FileReader fin = new FileReader(t_pattern);

        String x = fin.toString;
        String y = before;
        String z = after;
        String m = replace (x,y,z);
        System.out.println (m);

        FileWriter fout = new FileWriter(t_patfile);
        While ((n = fin.read (c) ) !=_1)
            Fout.write(c,0,n);
        Fin.close ( );
        Fout.close( );
    }
}
```

รูปที่ 4.23 รายละเอียดเมทธอดแพทเทิร์นทูโค้ดในคลาสแพทเทิร์นเจนเนอร์เรเตอร์

4.4 การสร้างแพ็กเกจสร้างชุดคำสั่ง

ในแพ็กเกจสร้างชุดคำสั่งจะมีคลาสโค้ดเจนเนอเรเตอร์ เป็นคลาสหลักในการสร้างชุดคำสั่งจากการใช้องค์ประกอบของซอฟต์แวร์ จากรูปที่ 3.8 คลาสแรทโหนด ประกอบไปด้วยเมทธอดต่างๆ ที่ใช้สร้างชุดคำสั่งได้แก่ เจนเนอร์เรทอินนิทโค้ด(generateInitCode) ทำหน้าที่สร้างชุดคำสั่งในส่วนโปรแกรมหลักเมื่อผู้ใช้สร้างโปรแกรมใหม่และทำการประกาศตัวแปร เมทธอดเจนเนอร์เรทอีเวนท์โค้ด(generateEventsCode) ทำหน้าที่สร้างชุดคำสั่งในส่วนเหตุการณ์ของออปเจคที่ใช้งานขณะออกแบบบนฟอร์มวินโดว์ เมทธอดเจนเนอร์เรทพรอบเพอร์ตี้โค้ด(generatePropertiesCode) จะสร้างชุดคำสั่งให้ เมื่อมีการแก้ไขค่าต่าง ๆ ในวินโดว์คุณสมบัติของออปเจคที่ใช้งาน รายละเอียดภายในคลาส แรทโหนด ดังแสดงในรูปที่ 4.24 และรายละเอียดเมทธอดเจนเนอร์เรทอีเวนท์โค้ดในคลาสแรทโหนด ดังแสดงในรูปที่ 4.25

RADNode
initializeFrom:RADNode gotoMethod:String bean:Object beanInfo:BeanInfo nodeProperties:Node$Property[] nodeEvents:Node$Property[] eventsList:EventsList defaultPropertyValues:Object[] newDeserializedBean:boolean eserializedOldEventHandlers:Stirng[] storedName:String changedProperties:Hashtable eventHandlers:Hashtable formManager:FormManager
formatName( ):String generateEventsCode(String,String):String generateInitCode(String):String generatePropertiesCode(String,Stirng):SystemAction[] getActions( ):Objcet getBean( ):Node$Property[] getBeanEvents( ):BeanInfo getBeanInfo( ):Node$Property[] getBeanProperties( ):String getCurrentName( ):Component getCustomizer( ):EventsList getEventsList( ):formManager getFormManager( ):RADNode getFormNode( ):String getName( ):Node$Property Set[] getPropertySet( ):String getSyntheticProperties( ):Node$Property[] init( ):void initializeEvents( ):void setBena(Object):void void setName(String) void setParentNode(Node)

รูปที่ 4.24 รายละเอียดภายในคลาสแรทโหนด

```
protected String generateEventsCode(String s, String s1)
  {
    StringBuffer stringbuffer = new StringBuffer();
    String s2 = FormUtils.getIndentString();
    EventsList.EventSet aeventset[] = eventsList.getEventSets();
    for(int i = 0; i < aeventset.length; i++)
    {
      EventsList.Event aevent[] = aeventset[i].getEvents();
      EventSetDescriptor eventsetdescriptor = aeventset[i].getEventSetDescriptor();
      Class class1 = FormUtils.getAdapterForListener(eventsetdescriptor.getListenerType());
      boolean flag = true;
      if(class1 == null)
      {
        class1 = eventsetdescriptor.getListenerType();
        flag = false;
      }
      boolean flag1 = false;
      boolean aflag[] = new boolean[aevent.length];
      for(int j = 0; j < aevent.length; j++)
        if(aevent[j].getHandler() != null)
        {
          flag1 = true;
          aflag[j] = true;
        }
        else
        {
          aflag[j] = false;
        }

      if(flag1 && !flag)
      {
        for(int k = 0; k < aevent.length; k++)
          aflag[k] = true;

      }
      if(flag1)
      {
        Method method = eventsetdescriptor.getAddListenerMethod();
        String s3 = s;
        boolean flag2 = false;
        if(method.getExceptionTypes().length == 1 && (class$java$util$TooManyListenersException
            == null ? (class$java$util$TooManyListenersException =
            class$("java.util.TooManyListenersException")) :
            class$java$util$TooManyListenersException).equals(method.getExceptionTypes()[0]))
            flag2 = true;
        if(flag2)
        {
          stringbuffer.append(s);
          stringbuffer.append("try {\n");
          s3 = s + s2;
        }
        stringbuffer.append(s3);
        stringbuffer.append(s1);
        stringbuffer.append(eventsetdescriptor.getAddListenerMethod().getName());
        stringbuffer.append(" (new ");
        stringbuffer.append(class1.getName() + " () {\n");
        for(int l = 0; l < aevent.length; l++)
      if(aflag[l])
      {
        Method method1 = aevent[l].getListenerMethod();
        Class aclass[] = method1.getParameterTypes();
        String as[];
      if(aclass.length == 1 && (class$java$util$EventObject == null ? (class$java$util$EventObject =
        class$("java.util.EventObject")) : class$java$util$EventObject).isAssignableFrom(aclass[0]))
      {
```

รูปที่ 4.25 รายละเอียดเมทธอดเจนเนอร์เรทอีเวนท์โค้ดในคลาสแรทโหนด

```
            as = (new String[] {
                FormEditor.getFormSettings().getEventVariableName()
            });
        }
        else
        {
            as = new String[aclass.length];
            for(int j1 = 0; j1 < aclass.length; j1++)
                as[j1] = "param" + j1;
                }
                stringbuffer.append(FormUtils.getMethodHeaderText(method1, s3 + s2 + s2, as));
                stringbuffer.append(" {\n");
                    if(aevent[l].getHandler() != null)
            {
            stringbuffer.append(s3);
            stringbuffer.append(s2);
            stringbuffer.append(s2);
            stringbuffer.append(s2);
            stringbuffer.append(aevent[l].getHandler().getName());
            stringbuffer.append(" (");
            for(int k1 = 0; k1 < as.length; k1++)
            {
                stringbuffer.append(as[k1]);
                if(k1 != as.length - 1)
                stringbuffer.append(", ");
                }
                stringbuffer.append(");");
            }
            stringbuffer.append("\n");
            stringbuffer.append(s3);
            stringbuffer.append(s2);
            stringbuffer.append(s2);
            stringbuffer.append("}\n");
            }
            stringbuffer.append(s3);
            stringbuffer.append(s2);
            stringbuffer.append("}\n");
            stringbuffer.append(s3);
            stringbuffer.append(");\n");
        if(flag2)
        {
        stringbuffer.append(s);
        stringbuffer.append("} catch (java.util.TooManyListenersException ");
        String s4 = "e";
        if(getFormManager().getVariablesPool().isReserved(s4))
        {
        int i1 = 1;
        for(s4 = "e1"; getFormManager().getVariablesPool().isReserved(s4); s4 = "e" + i1);
            }
            stringbuffer.append(s4);
            stringbuffer.append(") {\n");
            stringbuffer.append(s);
            stringbuffer.append(s2);
            stringbuffer.append(s4);
            stringbuffer.append(".printStackTrace ();\n");
            stringbuffer.append(s);
            stringbuffer.append("}\n");
            }
        }
    }
    return stringbuffer.toString();
}
```

รูปที่ 4.25 รายละเอียดเมทธอดเจนเนอร์เรทอีเวนท์โค้ดในคลาสแรทโหนด (ต่อ)