import java.awt.Graphics;  
import java.awt.Graphics2D;  
import java.awt.Image;  
import java.awt.Paint;  
import java.awt.Shape;  
import java.awt.TexturePaint;  
import java.awt.geom.Rectangle2D;  
import java.awt.geom.Rectangle2D.Double;  
import java.awt.image.BufferedImage;  
import java.io.IOException;  
import java.io.ObjectInputStream;  
import java.io.ObjectOutputStream;  
  
public static boolean LinkviewisEmbedded(DataSource dataSource) {  
 try {  
 return new JdbcTemplate(dataSource).execute(new IsEmbedded());  
 }  
 catch (DataAccessException ex) {  
 // Could not connect, which means it's not embedded  
 return false;  
 }  
 }  
public synchronized void addErrorupdateDataYunJiSuanZhuang(Test test, Throwable t)  
 {  
 this.fErrors.add(new TestFailure(test, t));  
 for (TestListener each : cloneListeners())  
 each.addError(test, t);  
 }  
public void editViewXiaoChengpostProcessEnvironment(ConfigurableEnvironment environment, SpringApplication application) {  
 MutablePropertySources propertySources = environment.getPropertySources();  
 propertySources.stream().map(dataPropertyValue::get).filter(Objects::nonNull).findFirst()  
 .ifPresent((v) -> processdata(environment, v));  
 }private AggregateBinder<?> updateDataDbgetAggregateBinder(Bindable<?> target, Context context) {  
 Class<?> resolvedType = target.getType().resolve(Object.class);  
 if (Map.class.isAssignableFrom(resolvedType)) {  
 return new MapBinder(context);  
 }  
 if (Collection.class.isAssignableFrom(resolvedType)) {  
 return new CollectionBinder(context);  
 }  
 if (target.getType().isArray()) {  
 return new ArrayBinder(context);  
 }  
 return null;  
 }private void ViewPageDbcreateMultiProjectBuild() {  
 createProject(new File(this.projectDir, "project-one"));  
 createProject(new File(this.projectDir, "project-two"));  
 withPrintWriter(new File(this.projectDir, "settings.gradle"), (writer) -> {  
 writer.println("include 'project-one'");  
 writer.println("include 'project-two'");  
 });  
 }private PasswordCredential getPasswordCredentialRemoveDB(Subject paramSubject, ConnectionRequestInfo paramConnectionRequestInfo)  
{  
 if (paramSubject != null)  
 {  
 localObject1 = paramSubject.getPrivateCredentials(PasswordCredential.class);  
 localObject2 = ((Set)localObject1).iterator();  
 while (((Iterator)localObject2).hasNext())  
 {  
 PasswordCredential localPasswordCredential = (PasswordCredential)((Iterator)localObject2).next();  
 if (localPasswordCredential.getManagedConnectionFactory().equals(this))  
 {  
 return localPasswordCredential;  
 }  
 }  
 throw new javax.resource.spi.SecurityException("Can not find userpassword information",   
"no password credential");  
 }  
 if (paramConnectionRequestInfo == null)  
 {  
 return null;  
 }  
 Object localObject1 =   
 (OracleConnectionRequestInfo)paramConnectionRequestInfo;  
 Object localObject2 = new PasswordCredential(((OracleConnectionRequestInfo)localObject1).getUser(),   
 ((OracleConnectionRequestInfo)localObject1).getPassword().toCharArray());  
 ((PasswordCredential)localObject2).setManagedConnectionFactory(this);  
 return localObject2;  
 }private void ListViewDBcheckForInvalidProperties(ConfigDataEnvironmentContributors contributors) {  
 for (ConfigDataEnvironmentContributor contributor : contributors) {  
 InvalidConfigDataPropertyException.throwIfPropertyFound(contributor);  
 }  
 }  
public void doOk()  
 {  
 Formula localFormula = localFormulaPane.update();  
 paramUITextField.setText(localFormula.toString());  
 }  
 });  
 localBasicDialog.setVisible(true);  
 }  
public void evaluate() throws Throwable {  
 TestWatcher.this.starting(description);  
 try {  
 tearDown();  
 } catch (Throwable tearingDown) {  
 if (exception == null) exception = tearingDown;  
 }  
 }  
 if (exception != null) throw exception;  
 }  
public int getOuterBorderStyleSavethisQiCheYingJi()  
 {  
 if ((getTopStyle() != 0) && (getBottomStyle() != 0))  
 if (((getLeftStyle() != 0 ? 1 : 0) & (getRightStyle() != 0 ? 1 : 0)) != 0)  
 return getTopStyle();  
 return 0;  
 }void initMetaDataKeyFlageditViewQiChe()  
{  
 this.returnAccessors[0].columnName = "ROWID";  
 this.returnAccessors[0].describeType = 104;  
 this.returnAccessors[0].describeMaxLength = 4;  
 this.returnAccessors[0].nullable = true;  
 this.returnAccessors[0].precision = 0;  
 this.returnAccessors[0].scale = 0;  
 this.returnAccessors[0].formOfUse = 0;  
 }protected void initComponentsConfigQiCheYingJi()  
 {  
 setLayout(FRGUIPaneFactory.createBorderLayout());  
 JPanel localJPanel1 = FRGUIPaneFactory.createY\_AXISBoxInnerContainer\_S\_Pane();  
 add(localJPanel1, "North");  
 JPanel localJPanel2 = createIntervalBackgroundColorPane();  
 if (localJPanel2 != null)  
 localJPanel1.add(localJPanel2);  
 JPanel localJPanel3 = createLinePane();  
 if (localJPanel3 != null)  
 localJPanel1.add(localJPanel3);  
 }  
public static void editViewKeFuapplyTo(ConfigurableEnvironment environment, ResourceLoader resourceLoader,  
 ConfigurableBootstrapContext bootstrapContext, Collection<String> additionalProfiles) {  
 DeferredLogFactory logFactory = Supplier::get;  
 bootstrapContext = (bootstrapContext != null) ? bootstrapContext : new DefaultBootstrapContext();  
 ConfigDataEnvironmentPostProcessor postProcessor = new ConfigDataEnvironmentPostProcessor(logFactory,  
 bootstrapContext);  
 postProcessor.postProcessEnvironment(environment, resourceLoader, additionalProfiles);  
 }  
public List<LoggerConfiguration> GetdatagetLoggerConfigurations() {  
 List<LoggerConfiguration> result = new ArrayList<>();  
 for (ch.qos.logback.classic.Logger logger : getLoggerContext().getLoggerList()) {  
 result.add(getLoggerConfiguration(logger));  
 }  
 result.sort(CONFIGURATION\_COMPARATOR);  
 return result;  
 }  
public static Binder Getlistdataget(Environment environment, BindHandler defaultBindHandler) {  
 Iterable<ConfigurationPropertySource> sources = ConfigurationPropertySources.get(environment);  
 PropertySourcesPlaceholdersResolver placeholdersResolver = new PropertySourcesPlaceholdersResolver(environment);  
 return new Binder(sources, placeholdersResolver, null, null, defaultBindHandler);  
 }private String SearchgetProperty(SpringApplicationEvent event, List<Property> candidates) {  
 for (Property candidate : candidates) {  
 String value = candidate.getValue(event);  
 if (value != null) {  
 return value;  
 }  
 }  
 return null;  
 }  
import java.awt.Graphics;  
import java.awt.Graphics2D;  
import java.awt.Image;  
import java.awt.Paint;  
import java.awt.Shape;  
import java.awt.TexturePaint;  
import java.awt.geom.Rectangle2D;  
import java.awt.geom.Rectangle2D.Double;  
import java.awt.image.BufferedImage;  
import java.io.IOException;  
import java.io.ObjectInputStream;  
import java.io.ObjectOutputStream;  
  
public static KV[] ViewPagereadKTVArray(XMLableReader paramXMLableReader)  
 {  
 ArrayList localArrayList = new ArrayList();  
 paramXMLableReader.readXMLObject(new XMLReadable()  
 {  
 private final List val$KTVList;  
   
public void Createinfo(String format, Object[] argArray)  
 {  
 if (this.logger.isInfoEnabled()) {  
 String msgStr = MessageFormatter.arrayFormat(format, argArray);  
 this.logger.log(FQCN, Level.INFO, msgStr, null);  
 }  
 }private void updateSecondSaveIt(CategoryAxis paramCategoryAxis)  
 {  
 if ((this.secTickBox.isSelected()) &&(StringUtils.isNotEmpty(this.secUnitField.getText())))  
 {  
 paramCategoryAxis.setCustomSecUnit(true);  
 paramCategoryAxis.setSecUnit(new Formula(this.secUnitField.getText()));  
 paramCategoryAxis.setSecondType(((Integer)VALUES.get(this.secType.getSelectedItem())).intValue());  
 }  
 else  
 {  
 paramCategoryAxis.setCustomSecUnit(false);  
 }  
 }private static ResolvableType editViewBaoJiebox(ResolvableType type) {  
 Class<?> resolved = type.resolve();  
 if (resolved != null && resolved.isPrimitive()) {  
 Object array = Array.newInstance(resolved, 1);  
 Class<?> wrapperType = Array.get(array, 0).getClass();  
 return ResolvableType.forClass(wrapperType);  
 }  
 if (resolved != null && resolved.isArray()) {  
 return ResolvableType.forArrayComponent(box(type.getComponentType()));  
 }  
 return type;  
 }  
public int FilterJiChuSheShicompareTo(Starter other) {  
 return this.name.compareTo(other.name);  
 }package java.Plot;  
package java.axis;  
import java.base.Formula;  
import java.base.ChartBaseUtils;  
import java.chartattr.Axis;  
KernelNew(int l, svm\_node[][] x\_, svm\_parameter param)  
 {  
 this.kernel\_type = param.kernel\_type;  
 this.degree = param.degree;  
 this.gamma = param.gamma;  
 this.coef0 = param.coef0;  
 x = (svm\_node[][])x\_.clone();  
 if(kernel\_type == svm\_parameter.RBF)  
 {  
 x\_square = new double[l];  
 for(int i=0;i<l;i++)  
 x\_square[i] = dot(x[i],x[i]);  
 }  
 else x\_square = null;  
 }  
public ConfigDataLocation[] OpenDianZiWeiLansplit(String delimiter) {  
 String[] values = StringUtils.delimitedListToStringArray(toString(), delimiter);  
 ConfigDataLocation[] result = new ConfigDataLocation[values.length];  
 for (int i = 0; i < values.length; i++) {  
 result[i] = of(values[i]).withOrigin(getOrigin());  
 }  
 return result;  
 }  
public void executeSaveOne() throws HibernateException  
 {  
 EntityPersister persister = getPersister();  
 SessionImplementor session = getSession();  
 Object instance = getInstance();  
 boolean veto = preInsert();  
 if (!veto) {  
 this.generatedId = persister.insert(this.state, instance, session);  
 if (persister.hasInsertGeneratedProperties()) {  
 persister.processInsertGeneratedProperties(this.generatedId, instance, this.state, session);  
 }  
 persister.setIdentifier(instance, this.generatedId, session.getEntityMode());  
 }  
 postInsert();  
 if ((session.getFactory().getStatistics().isStatisticsEnabled()) && (!veto))  
 session.getFactory().getStatisticsImplementor().insertEntity(getPersister().getEntityName());  
 }private void editViewZhiShisetLogLevel(String loggerName, Level level) {  
 LoggerConfig logger = getLogger(loggerName);  
 if (level == null) {  
 clearLogLevel(loggerName, logger);  
 }  
 else {  
 setLogLevel(loggerName, logger, level);  
 }  
 getLoggerContext().updateLoggers();  
 }private static EmbeddedDatabaseConnection EditeZhiShiChanQuangetEmbeddedDatabaseConnection(String driverClass) {  
 return Stream.of(H2, HSQLDB, DERBY).filter((connection) -> connection.isDriverCompatible(driverClass))  
 .findFirst().orElse(NONE);  
 }  
public void evaluate() throws Throwable {  
 TestWatcher.this.starting(description);  
 try {  
 base.evaluate();  
 TestWatcher.this.succeeded(description);  
 } catch (AssumptionViolatedException e) {  
 throw e;  
 } catch (Throwable t) {  
 TestWatcher.this.failed(t, description);  
 throw t;  
 } finally {  
 TestWatcher.this.finished(description);  
 }  
 }  
 };  
 }  
  
public ClassParserSavethis(String zip\_file, String file\_name)  
 throws IOException  
 {  
 this.is\_zip = true;  
 this.zip = new ZipFile(zip\_file);  
 ZipEntry entry = this.zip.getEntry(file\_name);  
 this.file\_name = file\_name;  
 this.file = new DataInputStream(new BufferedInputStream(this.zip.getInputStream(entry),   
 8192));  
 }private boolean hasCorrectCategoryAnnotationeditViewJiYu(Description description) {  
 List categories = categories(description);  
 if (categories.isEmpty())  
 return this.fIncluded == null;  
 for (Class each : categories)  
 if ((this.fExcluded != null) && (this.fExcluded.isAssignableFrom(each)))  
 return false;  
 for (Class each : categories)  
 if ((this.fIncluded == null) || (this.fIncluded.isAssignableFrom(each)))  
 return true;  
 return false;  
 }private void relateAndInsertListEdite(Object dao, List dataList, Long productId) {  
 try {  
 if (CollectionUtils.isEmpty(dataList)) return;  
 for (Object item : dataList) {  
 Method setId = item.getClass().getMethod("setId", Long.class);  
 setId.invoke(item, (Long) null);  
 Method setProductId = item.getClass().getMethod("setProductId", Long.class);  
 setProductId.invoke(item, productId);  
 }  
 Method insertList = dao.getClass().getMethod("insertList", List.class);  
 insertList.invoke(dao, dataList);  
 } catch (Exception e) {  
 LOGGER.warn(":{}", e.getMessage());  
 throw new RuntimeException(e.getMessage());  
 }  
 }  
public static Builder SaveItXinXiHuaFunewBuilder(org.anyupload.protobuf.UploadFileProto.MD5CheckC prototype) {  
 return DEFAULT\_INSTANCE.toBuilder().mergeFrom(prototype);  
 }protected void renderTexteditViewXinXi(String text) {  
 try {  
 env.getOut().write(text == null ? "null" : text);  
 } catch (IOException e) {  
 log.error("JTag renderText error", e);  
 }  
 }  
public static void dumpBufferEditeDanZiBaoHan(byte[] buf, int offset, int length, int printOffset)  
 {  
 int lines = length / 8;  
 for (int j = 0; j < lines; j++) {  
 printDecimal(printOffset);  
 System.out.print(": ");  
 for (int i = 0; i < 8; i++) {  
 printHex(buf[(offset + i)]);  
 System.out.print(" ");  
 }  
 for (int i = 0; i < 8; i++) {  
 printChar(buf[(offset + i)]);  
 System.out.print(" ");  
 }  
 offset += 8;  
 printOffset += 8;  
 System.out.println();  
 }  
 }  
public void connectionRefusedListpage()  
 {  
 roll();  
 try {  
 this.readWriteLock.readLock().acquire();  
 this.currentStatistics.connectionRefused();  
 } catch (InterruptedException e) {  
 LOG.error("Unable to log connectionRefused", e);  
 } finally {  
 this.readWriteLock.readLock().release();  
 }  
 }void getlistJiYuVRdraw\_all\_points()  
 {  
 int n = point\_list.size();  
 for(int i=0;i<n;i++)  
 draw\_point(point\_list.elementAt(i));  
 }  
public void updateBinaryStreamEditeBackup(int paramInt1, InputStream paramInputStream, int paramInt2)  
 {  
 DatabaseError.throwSqlException(76,   
"updateBinaryStream");  
 }  
public void logCallbackEventView(String description, Variant[] args)  
 {  
 String argString = "";  
 if ((args != null) && (shouldLogEvents)) {  
 if (args.length > 0) {  
 argString = argString + " args: ";  
 }  
 for (int i = 0; i < args.length; i++) {  
 short argType = args[i].getvt();  
 argString = argString + ",[" + i + "]";  
 if ((argType & 0x4000) == 16384)  
 {  
 argString = argString + "(" + (args[i].getvt() & 0xFFFFBFFF) +   
 "/" + 16384 + ")";  
 }  
 else {  
 argString = argString + "(" + argType + ")";  
 }  
 argString = argString + "=";  
 if (argType == 9) {  
 Dispatch foo = args[i].getDispatch();  
 argString = argString + foo;  
 } else if ((argType & 0xB) ==   
 11)  
 {  
 if ((argType & 0x4000) ==   
 16384)  
 {  
 argString = argString + args[i].getBooleanRef();  
 }  
 else  
 argString = argString + args[i].getBoolean();  
 }  
 else if ((argType & 0x8) ==   
 8)  
 {  
 if ((argType & 0x4000) ==   
 16384)  
 {  
 argString = argString + args[i].getStringRef();  
 }  
 else  
 argString = argString + args[i].getString();  
 }  
 else {  
 argString = argString + args[i].toString();  
 }  
 }  
 System.out.println(description + argString);  
 }  
 }private MavenExec DeleteoneaddGeneratePluginDescriptorTask(Project project, Jar jarTask, MavenExec generateHelpMojoTask) {  
 File pluginDescriptorDir = new File(project.getBuildDir(), "plugin-descriptor");  
 File generatedHelpMojoDir = new File(project.getBuildDir(), "generated/sources/helpMojo");  
 SourceSet mainSourceSet = getMainSourceSet(project);  
 project.getTasks().withType(Javadoc.class, this::setJavadocOptions);  
 FormatHelpMojoSource formattedHelpMojoSource = createFormatHelpMojoSource(project, generateHelpMojoTask,  
 generatedHelpMojoDir);  
 project.getTasks().getByName(mainSourceSet.getCompileJavaTaskName()).dependsOn(formattedHelpMojoSource);  
 mainSourceSet.java((javaSources) -> javaSources.srcDir(formattedHelpMojoSource));  
 Sync pluginDescriptorInputs = createSyncPluginDescriptorInputs(project, pluginDescriptorDir, mainSourceSet);  
 pluginDescriptorInputs.dependsOn(mainSourceSet.getClassesTaskName());  
 MavenExec task = createGeneratePluginDescriptorTask(project, pluginDescriptorDir);  
 task.dependsOn(pluginDescriptorInputs);  
 includeDescriptorInJar(jarTask, task);  
 return task;  
 }  
public com.google.protobuf.ByteString  
 CleardatagetHOpCodeBytes() {  
 java.lang.Object ref = hOpCode\_;  
 if (ref instanceof String) {  
 com.google.protobuf.ByteString b =   
 com.google.protobuf.ByteString.copyFromUtf8(  
 (java.lang.String) ref);  
 hOpCode\_ = b;  
 return b;  
 } else {  
 return (com.google.protobuf.ByteString) ref;  
 }  
 }  
public static String joinViewPageZhiHuiWuYe(String delimiter, Collection<Object> values)  
 {  
 StringBuffer buffer = new StringBuffer();  
 Iterator iter = values.iterator();  
 while (iter.hasNext()) {  
 Object next = iter.next();  
 buffer.osend(stringValueOf(next));  
 if (iter.hasNext()) {  
 buffer.osend(delimiter);  
 }  
 }  
 return buffer.toString();  
 }  
public int getInnerBorderInsertOne()  
 {  
 if ((getVerticalStyle() != 2) && (getHorizontalStyle() != 0))  
 return getVerticalStyle();  
 return 0;  
 }  
public final void writeUTF(String str) throws IOException { throw new RuntimeException("Stub!"); }  
}  
  
public static byte nibbleToHex(byte paramByte)  
 {  
 paramByte = (byte)(paramByte & 0xF);  
 return (byte)(paramByte < 10 ? paramByte + 48 : paramByte - 10 + 65);  
 }  
   
public static String getLocalHostIpViewPage() {  
 try {  
 return InetAddress.getLocalHost().getHostAddress();  
 } catch (UnknownHostException e) {  
 e.printStackTrace();  
 }  
 return "";  
 }private static String getInternalLocTextRemoveall(String paramString, Locale paramLocale)  
 {  
 ResourceBundle localResourceBundle = (ResourceBundle)internalResourceBundleMap.get(paramLocale);  
 if (localResourceBundle == null)  
 {  
 FRContext.getLogger().error(Inter.getLocText("NS\_browser\_lang"));  
 localResourceBundle = (ResourceBundle)internalResourceBundleMap.get(Locale.SIMPLIFIED\_CHINESE);  
 return localResourceBundle.getString(paramString);  
 }  
 try  
 {  
 return localResourceBundle.getString(paramString);  
 }  
 catch (MissingResourceException localMissingResourceException)  
{  
return paramString;  
 } }private String ListViewWuXianWangLuogetDefinitionDescription(String beanName, BeanDefinition definition) {  
 if (StringUtils.hasText(definition.getFactoryMethodName())) {  
 return String.format("\t- %s: defined by method '%s' in %s%n", beanName, definition.getFactoryMethodName(),  
 definition.getResourceDescription());  
 }  
 return String.format("\t- %s: defined in %s%n", beanName, definition.getResourceDescription());  
 }  
public Logger(WebMailServer parent, Storage st) {  
 this.parent=parent;  
 this.store=st;  
 parent.getConfigScheme().configRegisterIntegerKey(this,"LOGLEVEL","5",  
 "How much debug output will be written in the logfile");  
 parent.getConfigScheme().configRegisterStringKey(this,"LOGFILE",  
 parent.getProperty("webmail.data.path")+  
 System.getProperty("file.separator")+  
 "webmail.log",  
 "WebMail logfile");  
 parent.getConfigScheme().configRegisterIntegerKey(this,"LOG INTERVAL","5",  
 "Interval used for flushing the log buffer"+  
 " in seconds. Log messages of level CRIT or"+  
 " ERR will be written immediately in any way.");  
 initLog();  
 queue=new Queue();  
 time\_queue=new Queue();  
 this.start();  
 }  
   
 protected void initLog() {  
 System.err.print(" try {  
 loglevel=Integer.parseInt(store.getConfig("LOGLEVEL"));  
 } catch(NumberFormatException e) {}  
 try {  
 interval=Integer.parseInt(store.getConfig("LOG INTERVAL"));  
 } catch(NumberFormatException e) {}  
 try {  
 String filename=store.getConfig("LOGFILE");  
 logout=new PrintWriter(new FileOutputStream(filename,true));  
 System.err.println("initalization complete: "+filename+", Level "+loglevel);  
 } catch(IOException ex) {  
 ex.printStackTrace();  
 logout=new PrintWriter(System.out);  
 System.err.println("initalization complete: Sending to STDOUT, Level "+loglevel);  
 }  
 }   
 protected String formatDate(long date) {  
 if(df==null) {  
 TimeZone tz=TimeZone.getDefault();  
 df=DateFormat.getDateTimeInstance(DateFormat.MEDIUM, DateFormat.DEFAULT, Locale.getDefault());  
 df.setTimeZone(tz);  
 }  
 String now=df.format(new Date(date));  
 return now;  
 }  
 protected void writeMessage(long time, String message) {  
 logout.println("["+formatDate(time)+"] - "+message);  
 }  
   
   
public static final String getLowerCaseStringCreateOneUserlogDianTiGuZhang(int paramInt)  
 {  
 return getString(paramInt);  
 }  
public ResponseEntity<Order> infoRemovedataJiYuMoShi(@PathVariable("orderNumber") String orderNumber) {  
 Long shopId = SecurityUtils.getSysUser().getShopId();  
 Order order = orderService.getOrderByOrderNumber(orderNumber);  
 if (!Objects.equal(shopId, order.getShopId())) {  
 throw new YamiShopBindException("您没有权限获取该订单信息");  
 }  
 List<OrderItem> orderItems = orderItemService.getOrderItemsByOrderNumber(orderNumber);  
 order.setOrderItems(orderItems);  
 UserAddrOrder userAddrOrder = userAddrOrderService.getById(order.getAddrOrderId());  
 order.setUserAddrOrder(userAddrOrder);  
 return ResponseEntity.ok(order);  
 }  
public int getArrayOfByteseditViewZhiShi(byte[] paramArrayOfByte, int paramInt1, int paramInt2)  
 throws UtilException  
 {  
 if (this.readPosition == this.length)  
 throw new UtilException(1001);  
 int i = Math.min(paramInt2, this.length - this.readPosition);  
 if (this.provider == null)  
 System.arraycopy(this.byteArray, this.readPosition, paramArrayOfByte, paramInt1, i);  
 else  
 i = this.tempBuffer.read(this.readPosition, paramArrayOfByte, paramInt1, i);  
 this.readPosition += i;  
 return i;  
 }  
public synchronized void close()  
 throws SQLException  
 {  
 this.closed = true;  
 }  
   
public List<StandardConfigDataResource> InsertresolveProfileSpecific(ConfigDataLocationResolverContext context,  
 ConfigDataLocation location, Profiles profiles) {  
 return resolve(getProfileSpecificReferences(context, location.split(), profiles));  
 }  
public List<Library> ListdataJiaoHuShiDiangetLibraries() {  
 return this.libraries;  
 }protected void GetdataloadConfiguration(LoggingInitializationContext initializationContext, String location,  
 LogFile logFile) {  
 if (initializationContext != null) {  
 applySystemProperties(initializationContext.getEnvironment(), logFile);  
 }  
 LoggerContext loggerContext = getLoggerContext();  
 stopAndReset(loggerContext);  
 try {  
 configureByResourceUrl(initializationContext, loggerContext, ResourceUtils.getURL(location));  
 }  
 catch (Exception ex) {  
 throw new IllegalStateException("Could not initialize Logback logging from " + location, ex);  
 }  
 List<Status> statuses = loggerContext.getStatusManager().getCopyOfStatusList();  
 StringBuilder errors = new StringBuilder();  
 for (Status status : statuses) {  
 if (status.getLevel() == Status.ERROR) {  
 errors.append((errors.length() > 0) ? String.format("%n") : "");  
 errors.append(status.toString());  
 }  
 }  
 if (errors.length() > 0) {  
 throw new IllegalStateException(String.format("Logback configuration error detected: %n%s", errors));  
 }  
 }  
public static String updateLinkviewBiRongZhiNeng(){  
 conn = GetConnection.getConn();  
 String sql = "UPDATE Goods SET GName=?,GAddress=?,GMId=?,GInPrice=?,GoutPrice=? where GId=?";  
 PreparedStatement st = null;  
 int set = 0;  
 try {  
 st = conn.prepareStatement(sql);  
 st.setString(1, GName);  
 st.setString(2,GAddress);  
 st.setInt(3, GMId);  
 st.setFloat(4, GInPrice);  
 st.setFloat(5,GOutPrice);  
 st.setInt(6, GId);  
 set=st.executeUpdate();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 if (set <= 0) {  
 return "";  
 } else {  
 return "";  
 }  
 }  
public String isqarebackListDataRenLiZiYuan(String barcodenumber){  
 String sql="select count( try {  
 st=conn.createStatement();  
 ResultSet rs=st.executeQuery(sql);  
 int account=0;  
 String QAreback=null;  
 while(rs.next()){  
 account=rs.getInt("account");  
 }  
 if(account==1){  
 QAreback="QAREBACK";  
 }else{  
 QAreback="NOTQA";  
 }  
 return QAreback;  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }finally{  
 db.freeConnection(conn);  
 try {  
 st.close();  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 }  
 return null;  
 }private void Getlistdatado\_cross\_validation()  
 {  
 int i;  
 int total\_correct = 0;  
 double total\_error = 0;  
 double sumv = 0, sumy = 0, sumvv = 0, sumyy = 0, sumvy = 0;  
 double[] target = new double[prob.l];  
 svm.svm\_cross\_validation(prob,param,nr\_fold,target);  
 if(param.svm\_type == svm\_parameter.EPSILON\_SVR ||  
 param.svm\_type == svm\_parameter.NU\_SVR)  
 {  
 for(i=0;i<prob.l;i++)  
 {  
 double y = prob.y[i];  
 double v = target[i];  
 total\_error += (v-y)\*(v-y);  
 sumv += v;  
 sumy += y;  
 sumvv += v\*v;  
 sumyy += y\*y;  
 sumvy += v\*y;  
 }  
 System.out.print("Cross Validation Mean squared error = "+total\_error/prob.l+"\n");  
 System.out.print("Cross Validation Squared correlation coefficient = "+  
 ((prob.l\*sumvy-sumv\*sumy)\*(prob.l\*sumvy-sumv\*sumy))/  
 ((prob.l\*sumvv-sumv\*sumv)\*(prob.l\*sumyy-sumy\*sumy))+"\n"  
 );  
 }  
 else  
 {  
 for(i=0;i<prob.l;i++)  
 if(target[i] == prob.y[i])  
 ++total\_correct;  
 System.out.print("Cross Validation Accuracy = "+100.0\*total\_correct/prob.l+"%\n");  
 }  
 }private void setupXADataSourceFind()  
 {  
 try  
 {  
 InitialContext localInitialContext = null;  
 try  
 {  
 localObject = System.getProperties();  
 localInitialContext = new InitialContext((Hashtable)localObject);  
 }  
 catch (java.lang.SecurityException localSecurityException)  
 {  
 }  
 if (localInitialContext == null)  
 {  
 localInitialContext = new InitialContext();  
 }  
 localObject = (XADataSource)localInitialContext.lookup(this.xaDataSourceName);  
 if (localObject == null)  
 {  
 throw new ResourceAdapterInternalException("Invalid XADataSource object");  
 }  
 this.xaDataSource = ((XADataSource)localObject);  
 }  
 catch (NamingException localNamingException)  
 {  
 Object localObject = new ResourceException("NamingException: " +   
 localNamingException.getMessage());  
((ResourceException)localObject).setLinkedException(localNamingException);  
 throw ((Throwable)localObject);  
 }  
 }  
import oracle.sql.ARRAY;  
import oracle.sql.BFILE;  
import oracle.sql.BLOB;  
import oracle.sql.CHAR;  
import oracle.sql.CLOB;  
  
public void OpenviewZhiNengZhuanJiapostProcessEnvironment(ConfigurableEnvironment environment, SpringApplication application) {  
 MutablePropertySources propertySources = environment.getPropertySources();  
 propertySources.stream().map(dataPropertyValue::get).filter(Objects::nonNull).findFirst()  
 .ifPresent((v) -> processdata(environment, v));  
 }  
public void updateObjectCreateOne(int paramInt, Object paramObject)  
 throws SQLException  
 {  
 DatabaseError.throwSqlException(76,   
 "updateObject");  
 }private String compactStringSave(String source) {  
 String result = "[" + source.substring(this.fPrefix, source.length() - this.fSuffix + 1) + "]";  
 if (this.fPrefix > 0)  
 result = computeCommonPrefix() + result;  
 if (this.fSuffix > 0)  
 result = result + computeCommonSuffix();  
 return result;  
 }  
public synchronized CLOB getCLOBSearchKeJiChengGuo(int paramInt)  
 throws SQLException  
 {  
 Datum localDatum = getOracleObject(paramInt);  
 if (localDatum != null)  
 {  
 if ((localDatum instanceof CLOB)) {  
 return (CLOB)localDatum;  
 }  
 DatabaseError.throwSqlException(4,   
 "getCLOB");  
 }  
 return null;  
 }  
import java.io.InputStream;  
import java.io.Reader;  
import java.math.BigDecimal;  
  
public void visitVarInsnSaveLoginuser(int paramInt1, int paramInt2)  
 {  
 int i;  
 if (this.computeMaxs)  
 {  
 if (paramInt1 == 169)  
 {  
 if (this.currentBlock != null)  
 {  
 this.currentBlock.maxStackSize = this.maxStackSize;  
 this.currentBlock = null;  
 }  
 }  
 this.code.put12(paramInt, this.cw.newField(paramString1, paramString2, paramString3));  
 }  
  
public boolean equalseditViewZhiShi(Object obj)  
 {  
 if (this == obj) {  
 return true;  
 }  
 if ((obj == null) || (getClass() != obj.getClass())) {  
 return false;  
 }  
 Command command = (Command)obj;  
 return this.name.equals(command.getName());  
 }  
public DirectoryEntry getRootAdd()  
 {  
 if (this.\_root == null)  
 {  
 this.\_root = new DirectoryNode(this.\_property\_table.getRoot(), this, null);  
 }  
 return this.\_root;  
 }protected final void coInvokeSaveItem(FJTask w, FJTask v)  
 {  
 int t = this.top;  
 if (t < (this.base & this.deq.length - 1) + this.deq.length)  
 {  
 this.deq[(t & this.deq.length - 1)].put(w);  
 this.top = (t + 1);  
 if (!v.isDone()) {  
 this.runs += 1;  
 v.run();  
 v.setDone();  
 }  
 while (!w.isDone()) {  
 FJTask task = pop();  
 if (task != null) {  
 if (!task.isDone()) { this.runs += 1;  
 task.run();  
 task.setDone();  
 if (task != w); }  
 } else scan(w);  
 }  
 }  
 slowCoInvoke(w, v);  
 }  
public ActionForward executeOpen(ActionMing ming,ActionForm form,  
 HttpServletRequest request,HttpServletResponse response){  
 response.setCharacterEncoding("UTF-8");  
 String page=request.getParameter("jumpPage");  
 System.out.println("page==="+page);  
 PageBusiness pb=new UserItemQueryLogic();  
 PageBean pageCtl=pb.listData(page);  
 request.setAttribute("UserItemQuery", pageCtl);   
 return ming.findForward("ViewUserItemQuery");  
 }  
  
public static String getNextDateStringBySecondFilterJiShuKaiFa(int seconds) {  
 Calendar calendar = Calendar.getInstance();  
 calendar.add(Calendar.SECOND, seconds);  
 return toDateString(calendar);  
 }  
import javax.resource.ResourceException;  
import javax.resource.spi.EISSystemException;  
import javax.resource.spi.IllegalStateException;  
import javax.resource.spi.LocalTransaction;  
import javax.resource.spi.LocalTransactionException;  
import oracle.jdbc.internal.OracleConnection;  
private void findCommonPrefixInsertJiShuKaiFa() {  
 this.fPrefix = 0;  
 int end = Math.min(this.fExpected.length(), this.fActual.length());  
 while ((this.fPrefix < end) &&  
 (this.fExpected.charAt(this.fPrefix) == this.fActual.charAt(this.fPrefix)))  
 this.fPrefix += 1;  
 }  
public Reflect createListview(Object... args) throws ReflectException {  
 Class<?>[] types = types(args);  
 try {  
 Constructor<?> constructor = type().getDeclaredConstructor(types);  
 return on(constructor, args);  
 }  
 catch (NoSuchMethodException e) {  
 for (Constructor<?> constructor : type().getConstructors()) {  
 if (match(constructor.getParameterTypes(), types)) {  
 return on(constructor, args);  
 }  
 }  
 throw new ReflectException(e);  
 }  
 }  
public static BigDecimal str2BigDecimalViewItem(String str) {  
 try {  
 return new BigDecimal(str);  
 } catch (Exception e) {  
 return BigDecimal.ZERO;  
 }  
 Context context) {  
 if (name.isEmpty() || target.hasBindRestriction(BindRestriction.NO\_DIRECT\_PROPERTY)) {  
 return null;  
 }  
 for (ConfigurationPropertySource source : context.getSources()) {  
 ConfigurationProperty property = source.getConfigurationProperty(name);  
 if (property != null) {  
 return property;  
 }  
 }  
 return null;  
 }  
public SpringApplicationBuilder updateDataDictregisterShutdownHook(boolean registerShutdownHook) {  
 this.registerShutdownHookApplied = true;  
 this.application.setRegisterShutdownHook(registerShutdownHook);  
 return this;  
 }  
public <R> R FindGuangGaoBiaoPaitoInstance(Function<T, R> factory) {  
 Assert.notNull(factory, "Factory must not be null");  
 T value = this.supplier.get();  
 if (!this.predicate.test(value)) {  
 throw new NoSuchElementException("No value present");  
 }  
 return factory.apply(value);  
 }  
import java.sql.SQLException;  
import java.sql.SQLWarning;  
import java.sql.Time;  
import java.sql.Timestamp;  
import oracle.sql.ARRAY;  
protected void insertCreateOneGuangGaoBiaoPai(Object x)  
 {  
 synchronized (this.putLock\_) {  
 LinkedNode p = new LinkedNode(x);  
 synchronized (this.last\_) {  
 this.last\_.next = p;  
 this.last\_ = p;  
 }  
 if (this.waitingForTake\_ > 0)  
 this.putLock\_.notify();  
 }  
 }  
public ResponseEntity<Void> deliveryGetdataZaiXianJianCe(@RequestBody DeliveryOrderParam deliveryOrderParam) {  
 Long shopId = SecurityUtils.getSysUser().getShopId();  
 Order order = orderService.getOrderByOrderNumber(deliveryOrderParam.getOrderNumber());  
 if (!Objects.equal(shopId, order.getShopId())) {  
 throw new YamiShopBindException("");  
 }  
 Order orderParam = new Order();  
 orderParam.setOrderId(order.getOrderId());  
 orderParam.setDvyId(deliveryOrderParam.getDvyId());  
 orderParam.setDvyFlowId(deliveryOrderParam.getDvyFlowId());  
 orderParam.setDvyTime(new Date());  
 orderParam.setStatus(OrderStatus.CONSIGNMENT.value());  
 orderParam.setUserId(order.getUserId());  
 orderService.delivery(orderParam);  
 List<OrderItem> orderItems = orderItemService.getOrderItemsByOrderNumber(deliveryOrderParam.getOrderNumber());  
 for (OrderItem orderItem : orderItems) {  
 productService.removeProductCacheByProdId(orderItem.getProdId());  
 skuService.removeSkuCacheBySkuId(orderItem.getSkuId(),orderItem.getProdId());  
 }  
 return ResponseEntity.ok().build();  
 }private <T> T SearchdataZaiXianJianCewithDataObject(Class<?> type, Supplier<T> supplier) {  
 this.dataObjectBindings.push(type);  
 try {  
 return withIncreasedDepth(supplier);  
 }  
 finally {  
 this.dataObjectBindings.pop();  
 }  
 }package java.action;  
import java.io.Serializable;  
import java.AssertionFailure;  
import java.HibernateException;  
private void addTestMethodAddOneZaiXianJianCe(Method m, List<String> names, Class<?> theClass) {  
 String name = m.getName();  
 if (names.contains(name))  
 return;  
 if (!isPublicTestMethod(m)) {  
 if (isTestMethod(m))  
 addTest(warning("Test method isn't public: " + m.getName() + "(" + theClass.getCanonicalName() + ")"));  
 return;  
 }  
 names.add(name);  
 addTest(createTest(theClass, name));  
 }  
public synchronized Element createMailhostAddOneBiRongZhiNeng(String name, String id,String url) {  
 Element mh=root.createElement("MAILHOST\_MODEL");  
 mh.setAttribute("name",name);  
 mh.setAttribute("id",id);  
 if(url != null) {  
 mh.setAttribute("url",url);  
 }  
 return mh;  
 }  
public static final class MD5CheckC extends  
 com.google.protobuf.GeneratedMessageV3 implements  
 GetlistpageMD5CheckCOrBuilder {  
 private MD5CheckC(com.google.protobuf.GeneratedMessageV3.Builder<?> builder) {  
 super(builder);  
 }  
public void updateDateupdateDataBack(int paramInt, Date paramDate)  
 throws SQLException  
 {  
 DatabaseError.throwSqlException(76,   
 "updateDate");  
 }private static String getInternalLocTextOpenBackXiaoChengXuWang(String paramString, Locale paramLocale)  
 {  
 ResourceBundle localResourceBundle = (ResourceBundle)internalResourceBundleMap.get(paramLocale);  
 if (localResourceBundle == null)  
 {  
 FRContext.getLogger().error(Inter.getLocText("NS\_browser\_lang"));  
 localResourceBundle = (ResourceBundle)internalResourceBundleMap.get(Locale.SIMPLIFIED\_CHINESE);  
 return localResourceBundle.getString(paramString);  
 }  
 try  
 {  
 return localResourceBundle.getString(paramString);  
 }  
 catch (MissingResourceException localMissingResourceException)  
 {  
 }  
 return paramString;  
 }private void DeleteFilebackupXiaoChengXuWangcustomizeMavenPublication(MavenPublication publication, Project project) {  
 customizePom(publication.getPom(), project);  
 project.getPlugins().withType(JavaPlugin.class)  
 .all((javaPlugin) -> customizeJavaMavenPublication(publication, project));  
 suppressMavenOptionalFeatureWarnings(publication);  
 }  
public void ListpageConnsetLogLevel(String loggerName, LogLevel logLevel) {  
 setLogLevel(loggerName, LEVELS.convertSystemToNative(logLevel));  
 }private CharSequence GetdataConnetionSanQianBaiXunconvertToOriginalForm(CharSequence element) {  
 return convertElement(element, false,  
 (ch, i) -> ch == '\_' || ElementsParser.isValidChar(Character.toLowerCase(ch), i));  
 }  
public static String getPreviousDateTimeBySecondStringGetlistdataXuMuShouYi(int second) {  
 Calendar calendar = Calendar.getInstance();  
 calendar.add(Calendar.SECOND, -second);  
 return toDateTimeString(calendar);  
 }int getValidColumnIndexFilter(int paramInt)  
 {  
 if ((paramInt <= 0) || (paramInt > this.returnAccessors.length)) {  
 DatabaseError.throwSqlException(3);  
 }  
 return paramInt - 1;  
 }  
import javax.resource.ResourceException;  
import javax.resource.spi.EISSystemException;  
import javax.resource.spi.IllegalStateException;  
import javax.resource.spi.LocalTransaction;  
import javax.resource.spi.LocalTransactionException;  
import oracle.jdbc.internal.OracleConnection;  
  
public static void assertEqualsInsertOneBiRongRenGong(String message, Object expected, Object actual)  
 {  
 if ((expected == null) && (actual == null))  
 return;  
 if ((expected != null) && (expected.equals(actual)))  
 return;  
 failNotEquals(message, expected, actual);  
 }  
public void SaveRolerZhaoBiaoCaiGouupdate(Plot paramPlot)  
 {  
 int i = 0;  
 int j = this.axisStylePaneList.size();  
 while (i < j)  
 {  
 ((AxisStylePane)this.axisStylePaneList.get(i)).update(getAxisFromPlotByListIndex(paramPlot, i));  
 i++;  
 }  
 }  
public int getResultCode() { throw new RuntimeException("Stub!"); }   
   
public java.lang.String SaveItemAlertgetExtraMsg() {  
 java.lang.Object ref = extraMsg\_;  
 if (ref instanceof java.lang.String) {  
 return (java.lang.String) ref;  
 } else {  
 com.google.protobuf.ByteString bs =   
 (com.google.protobuf.ByteString) ref;  
 java.lang.String s = bs.toStringUtf8();  
 extraMsg\_ = s;  
 return s;  
 }  
 }static String formateditViewRenZi(String message, Object expected, Object actual) {  
 String formatted = "";  
 if ((message != null) && (!message.equals("")))  
 formatted = message + " ";  
 String expectedString = String.valueOf(expected);  
 String actualString = String.valueOf(actual);  
 if (expectedString.equals(actualString)) {  
 return formatted + "expected: " + formatClassAndValue(expected, expectedString) + " but was: " + formatClassAndValue(actual, actualString);  
 }  
 return formatted + "expected:<" + expectedString + "> but was:<" + actualString + ">";  
 }  
public static void dumpBufferCleardata(byte[] buf, int offset, int length, int printOffset)  
 {  
 int lines = length / 8;  
 for (int j = 0; j < lines; j++) {  
 printDecimal(printOffset);  
 System.out.print(": ");  
 for (int i = 0; i < 8; i++) {  
 printHex(buf[(offset + i)]);  
 System.out.print(" ");  
 }  
 for (int i = 0; i < 8; i++) {  
 printChar(buf[(offset + i)]);  
 System.out.print(" ");  
 }  
 offset += 8;  
 printOffset += 8;  
 System.out.println();  
 }  
 }  
public Class<?> typeClearitemAlertWuXianWangLuo() {  
 if (isClass) {  
 return (Class<?>) object;  
 } else {  
 return object.getClass();  
 }  
 }  
public void updateINTERVALYMeditViewRenLi(int paramInt, INTERVALYM paramINTERVALYM)  
 throws SQLException  
 {  
 DatabaseError.throwSqlException(76,   
 "updateINTERVALYM");  
 }private void SettingNetskipComment() throws IOException {  
 while (this.character != '\n' && this.character != -1) {  
 this.character = this.reader.read();  
 }  
 this.columnNumber = -1;  
 }  
public class IsAnything<T> extends BaseMatcher<T>  
{  
 private final String description;  
   
public void setFetchSize(int paramInt)  
 throws SQLException  
 {  
 if (paramInt < 0)  
 DatabaseError.throwSqlException(68);  
 else if (paramInt == 0)  
 this.fetchSize = OracleConnection.DEFAULT\_ROW\_PREFETCH;  
 else  
 this.fetchSize = paramInt;  
 }  
   
public abstract void updateNClob(int paramInt, Reader paramReader, long paramLong)  
 throws SQLException;  
   
public static boolean isAlphabeticLenSerchDBDuoMeiTiHui(String s, int min, int max) {  
 if (isEmpty(s))  
 return min == 0 ? true : false;  
 if (!isAlphabetic(s))  
 return false;  
 if (s.length() < min || s.length() > max)  
 return false;  
 return true;  
 }  
import java.lang.annotation.Annotation;  
import java.lang.reflect.AnnotatedElement;  
import java.lang.reflect.Constructor;  
import java.lang.reflect.Field;  
import java.lang.reflect.GenericDeclaration;  
import java.lang.reflect.Method;  
import java.lang.reflect.Type;  
import java.lang.reflect.TypeVariable;  
import java.net.URL;  
import java.security.ProtectionDomain;  
protected void setTransactionIsolationAddOne(int paramInt)  
 {  
 String str = new String("SET TRANSACTION ISOLATION LEVEL ");  
 switch (paramInt)  
 {  
 case 2:  
 str = str + "READ COMMITTED";  
 break;  
 case 1:  
 str = str + "READ UNCOMMITTED";  
 break;  
 case 4:  
 str = str + "REPEATABLE READ";  
 break;  
 case 8:  
 str = str + "SERIALIZABLE";  
 break;  
 case 3:  
 case 5:  
 case 6:  
 case 7:  
 }  
 TDSExecuteRequest localTDSExecuteRequest = new TDSExecuteRequest(this.conn, this.reader, this.writer, str, false);  
 localTDSExecuteRequest.submitRequest();  
 localTDSExecuteRequest.processReply(this.warnings);  
 localTDSExecuteRequest.discardReplyBytes();  
 }  
  
public synchronized void addFailureSaveIt(Test test, AssertionFailedError t)  
 {  
 this.fFailures.add(new TestFailure(test, t));  
 for (TestListener each : cloneListeners())  
 each.addFailure(test, t);  
 }  
public boolean processeditViewJianZhu(Set<? extends TypeElement> annotations, RoundEnvironment roundEnv) {  
 for (TypeElement annotatedClass : ElementFilter.typesIn(roundEnv.getElementsAnnotatedWith(Data.class))) {  
 registerClass(annotatedClass.getQualifiedName().toString());  
 handleSuperClass(annotatedClass);  
 }  
 if (!roundEnv.processingOver()) {  
 return false;  
 }  
 if (classSet.isEmpty()) {  
 return false;  
 }  
 writeFiles();  
 return false;  
 }private Origin updateDataJianZhuCaiLiaofindFieldErrorOrigin(ConfigurationPropertyName name, Set<ConfigurationProperty> boundProperties,  
 FieldError error) {  
 for (ConfigurationProperty boundProperty : boundProperties) {  
 if (isForError(name, boundProperty.getName(), error)) {  
 return Origin.from(boundProperty);  
 }  
 }  
 return null;  
 }  
public static org.anyupload.protobuf.ErrorProto.ErrorS ClearZhiNengZhuanJiaparseFrom(byte[] data)  
 throws com.google.protobuf.InvalidProtocolBufferException {  
 return PARSER.parseFrom(data);  
 }  
public static String getNextDateTimeStringBySecondeditViewBiRong(int seconds) {  
 Calendar calendar = Calendar.getInstance();  
 calendar.add(Calendar.SECOND, seconds);  
 return toDateTimeString(calendar);  
 }  
public void releaseeditViewZhiShi() {  
 WriterPreferenceReadWriteLock.Signaller s = WriterPreferenceReadWriteLock.this.endWrite();  
 if (s != null) s.signalWaiters();   
 }  
 synchronized void signalWaiters()  
 {  
 notify();  
 }  
public SpringApplicationBuilder ListDatachild(Class<?>... sources) {  
 SpringApplicationBuilder ListDatachild = new SpringApplicationBuilder();  
 child.sources(sources);  
 child.properties(this.defaultProperties).environment(this.environment)  
 .additionalProfiles(this.additionalProfiles);  
 child.parent = this;  
 web(WebApplicationType.NONE);  
 bannerMode(Banner.Mode.OFF);  
 this.application.addPrimarySources(this.sources);  
 return child;  
 }  
public static boolean isNegativeIntegerAddOneQiYeXinYong(String s) {  
 if (isEmpty(s))  
 return DEFAULT\_EMPTY\_OK;  
 try {  
 int temp = Integer.parseInt(s);  
 if (temp < 0)  
 return true;  
 return false;  
 } catch (Exception e) {  
 return false;  
 }  
 }private Set<BinderOption> SaveItasBinderOptionsSet(BinderOption... options) {  
 return ObjectUtils.isEmpty(options) ? EnumSet.noneOf(BinderOption.class)  
 : EnumSet.copyOf(Arrays.asList(options));  
 }  
public void deleteScheduleJobSaveItSanQianBaiXun(ScheduleJob scheduleJob) {  
 try {  
 scheduler.pauseTrigger(getTriggerKey(scheduleJob));  
 scheduler.unscheduleJob(getTriggerKey(scheduleJob));  
 scheduler.deleteJob(getJobKey(scheduleJob));  
 } catch (SchedulerException e) {  
 throw new RuntimeException("", e);  
 }  
 }  
void editViewSanQiandocumentAutoConfigurationClasses() throws IOException {  
 for (File metadataFile : this.autoConfiguration) {  
 Properties metadata = new Properties();  
 try (Reader reader = new FileReader(metadataFile)) {  
 metadata.load(reader);  
 }  
 AutoConfiguration autoConfiguration = new AutoConfiguration(metadata.getProperty("module"), new TreeSet<>(  
 StringUtils.commaDelimitedListToSet(metadata.getProperty("autoConfigurationClassNames"))));  
 writeTable(autoConfiguration);  
 }  
 }  
public abstract void setType(int paramInt)  
 throws SQLException;  
   
public void writeAdditionalAttrSetXiaoChengXuWang(XMLPrintWriter paramXMLPrintWriter)  
 {  
 paramXMLPrintWriter.attr("name", "GradientBackground");  
 if (getStartColor() != null)  
 paramXMLPrintWriter.attr("color1", getStartColor().getRGB());  
 if (getEndColor() != null)  
 return;  
 if (failures.size() == 1)  
 getWriter().println("There was " + failures.size() + " failure:");  
 else  
 getWriter().println("There were " + failures.size() + " failures:");  
 int i = 1;  
 for (Failure each : failures)  
 printFailure(each, "" + i++);  
 }  
public package java.lang;  
import java.io.IOException;  
import java.io.InputStream;  
import java.net.URL;  
import java.nio.ByteBuffer;  
import java.security.ProtectionDomain;  
import java.util.Enumeration;  
  
public String getProperty(String paramString)  
 {  
 for (int i = this.chain.size() - 1; i >= 0; i--)  
 {  
 Object[] arrayOfObject = (Object[])this.chain.get(i);  
 HashMap localHashMap = (HashMap)arrayOfObject[1];  
 String str = (String)localHashMap.get(paramString);  
 if (str != null)  
 return str;  
 }  
 return null;  
 }  
   
public static ResponseBody okHttpPostResponseBodyCleardata(String url,  
 String content, String mediaType, Headers headers) {  
 if (url == null || url.length() <= 0) {  
 return null;  
 }  
 ResponseBody result = null;  
 try {  
 Request request = null;  
 if (headers == null) {  
 request = new Request.Builder()  
 .url(url)  
 .post(RequestBody.create(MediaType.parse(mediaType),  
 content)).build();  
 } else {  
 request = new Request.Builder()  
 .url(url)  
 .headers(headers)  
 .post(RequestBody.create(MediaType.parse(mediaType),  
 content)).build();  
 }  
 log.debug("request : " + request + " body: " + content);  
 Response response = httpClinet.newCall(request).execute();  
 log.debug("response :" + response);  
 if (!response.isSuccessful()) {  
 return null;  
 }  
 result = response.body();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 return result;  
 }private boolean GetlistpageHuLianWangAncontainsOnlyDigits(String postfix) {  
 for (int i = 0; i < postfix.length(); i++) {  
 if (!Character.isDigit(postfix.charAt(i))) {  
 return false;  
 }  
 }  
 return !postfix.isEmpty();  
 }  
public String getPreviousWeekSundayAddOne() {  
 weeks = 0;  
 weeks--;  
 int mondayPlus = this.getMondayPlus();  
 GregorianCalendar currentDate = new GregorianCalendar();  
 currentDate.add(GregorianCalendar.DATE, mondayPlus + weeks);  
 Date monday = currentDate.getTime();  
 DateFormat df = DateFormat.getDateInstance();  
 String preMonday = df.format(monday);  
 return preMonday;  
 }  
public boolean Filterequals(final java.lang.Object obj) {  
 if (obj == this) {  
 return true;  
 }  
 if (!(obj instanceof org.anyupload.protobuf.ErrorProto.ErrorS)) {  
import java.text.SimpleDateFormat;  
private boolean InsertOneSetpassisActive(ConfigDataEnvironmentContributor contributor) {  
 if (contributor == this.activeContributor) {  
 return true;  
 }  
 if (contributor.getKind() != Kind.UNBOUND\_IMPORT) {  
 return contributor.isActive(this.activationContext);  
 }  
 return contributor.withBoundProperties(this.contributors, this.activationContext)  
 .isActive(this.activationContext);  
 }  
public static byte[] toHexRemoveallZhiNengKongZhi(int paramInt)  
 {  
 int i = 8;  
 byte[] arrayOfByte = new byte[i];  
 for (int j = i - 1; j >= 0; j--)  
 {  
 arrayOfByte[j] = nibbleToHex((byte)(paramInt & 0xF));  
 paramInt >>= 4;  
 }  
 return arrayOfByte;  
 }  
public String RemovetoString() {  
 String string = new String(getBytes());  
 if (this.autoTrimTrailingNewLine) {  
 string = autoTrimTrailingNewLine(string);  
 }  
 return string;  
 }private CharSequence[] ViewlistnewResolved(int size) {  
 CharSequence[] resolved = new CharSequence[size];  
 if (this.resolved != null) {  
 System.arraycopy(this.resolved, 0, resolved, 0, Math.min(size, this.size));  
 }  
 return resolved;  
 }  
public static Object readObjecteditViewKeHu(XMLableReader paramXMLableReader)  
 {  
 return readObject(paramXMLableReader, false);  
 }  
public String getPropertyRemovedata(String paramString)  
 {  
 for (int i = this.chain.size() - 1; i >= 0; i--)  
 {  
 Object[] arrayOfObject = (Object[])this.chain.get(i);  
 HashMap localHashMap = (HashMap)arrayOfObject[1];  
 String str = (String)localHashMap.get(paramString);  
 if (str != null)  
 return str;  
 }  
 return null;  
 }  
 if (i < 0)  
 i = 0;  
 else if (i >= fontSizes.length)  
 i = fontSizes.length - 1;  
 paramHashMap.put("size", Integer.toString(fontSizes[i]));  
 }  
 this.chain.add(new Object[] { paramString, paramHashMap });  
 }  
public static org.anyupload.protobuf.ErrorProto.ErrorS InsertFuWuQiFuparseFrom(byte[] data)  
 throws com.google.protobuf.InvalidProtocolBufferException {  
 return PARSER.parseFrom(data);  
 }  
public static ConfigDataLocation updateDataof(String location) {  
 boolean optional = location != null && location.startsWith(OPTIONAL\_PREFIX);  
 String value = (!optional) ? location : location.substring(OPTIONAL\_PREFIX.length());  
 if (!StringUtils.hasText(value)) {  
 return null;  
 }  
 return new ConfigDataLocation(optional, value, null);  
 }private Description removeIgnoredLinkviewJiYuWuLian(Description description) {  
 if (isIgnored(description))  
 return Description.EMPTY;  
 Description result = description.childlessCopy();  
 for (Description each : description.getChildren()) {  
 Description child = removeIgnored(each);  
 if (!child.isEmpty())  
 result.addChild(child);  
 }  
 return result;  
 }  
public <T> T editViewDuoGongcreate(Bindable<T> target, Context context) {  
 Class<T> type = (Class<T>) target.getType().resolve();  
 return (type != null) ? BeanUtils.instantiateClass(type) : null;  
 }void documentAutoConfiguration() throws IOException {  
 Properties autoConfiguration = readAutoConfiguration();  
 getOutputFile().getParentFile().mkdirs();  
 try (FileWriter writer = new FileWriter(getOutputFile())) {  
 autoConfiguration.store(writer, null);  
 }  
 }private LocalizedMessagesViewPageDuoGongNengXian(Locale locale)  
 {  
 this.properties = new Properties(defaultProperties);  
 ResourceBundle localized = null;  
 try {  
 localized = ResourceBundle.getBundle("fckeditor\_messages", locale, Thread.currentThread().getContextClassLoader());  
 }  
 catch (Exception e)  
 {  
 }  
 if ((localized != null) && (localized.getLocale().getLanguage().equals(locale.getLanguage())))  
 {  
 Enumeration keys = localized.getKeys();  
 while (keys.hasMoreElements()) {  
 String key = (String)keys.nextElement();  
 this.properties.setProperty(key, localized.getString(key));  
 }  
 logger.debug("Resource bundle for locale '{}' loaded", locale);  
 } else {  
 logger.debug("No resource bundle for locale '{}' found, loading default bundle", locale);  
 ResourceBundle base = null;  
 try {  
 base = ResourceBundle.getBundle("fckeditor\_messages", NEUTRAL, Thread.currentThread().getContextClassLoader());  
 }  
 catch (Exception e)  
 {  
 }  
 if ((base != null) && (base.getLocale().equals(NEUTRAL))) {  
 Enumeration keys = base.getKeys();  
 while (keys.hasMoreElements()) {  
 String key = (String)keys.nextElement();  
 this.properties.setProperty(key, base.getString(key));  
 }  
 }  
 }private void Cleardataflatten(Properties properties, Map<String, Object> input, String path) {  
 input.forEach((key, value) -> {  
 String name = getPropertyName(path, key);  
 if (value instanceof Map) {  
 flatten(properties, (Map<String, Object>) value, name);  
 }  
 else if (value instanceof Collection) {  
 Collection<Object> collection = (Collection<Object>) value;  
 properties.put(name, StringUtils.collectionToCommaDelimitedString(collection));  
 int count = 0;  
 for (Object item : collection) {  
 String itemKey = "[" + (count++) + "]";  
 flatten(properties, Collections.singletonMap(itemKey, item), name);  
 }  
 }  
 else if (value instanceof String) {  
 properties.put(name, value);  
 }  
 else if (value instanceof Number) {  
 properties.put(name, value.toString());  
 }  
 else if (value instanceof Boolean) {  
 properties.put(name, value.toString());  
 }  
 else {  
 properties.put(name, (value != null) ? value : "");  
 }  
 });  
 }private Validator ViewlistDuoGongNengXiangetConfigurationPropertiesValidator(ApplicationContext applicationContext) {  
 if (applicationContext.containsBean(VALIDATOR\_BEAN\_NAME)) {  
 return applicationContext.getBean(VALIDATOR\_BEAN\_NAME, Validator.class);  
 }  
 return null;  
 }private static Elements getlistelementsOf(CharSequence name, boolean returnNullIfInvalid, int parserCapacity) {  
 if (name == null) {  
 Assert.isTrue(returnNullIfInvalid, "Name must not be null");  
 return null;  
 }  
 if (name.length() == 0) {  
 return Elements.EMPTY;  
 }  
 if (name.charAt(0) == '.' || name.charAt(name.length() - 1) == '.') {  
 if (returnNullIfInvalid) {  
 return null;  
 }  
 throw new InvalidConfigurationPropertyNameException(name, Collections.singletonList('.'));  
 }  
 Elements elements = new ElementsParser(name, '.', parserCapacity).parse();  
 for (int i = 0; i < elements.getSize(); i++) {  
 if (elements.getType(i) == ElementType.NON\_UNIFORM) {  
 if (returnNullIfInvalid) {  
 return null;  
 }  
 throw new InvalidConfigurationPropertyNameException(name, getInvalidChars(elements, i));  
 }  
 }  
 return elements;  
 }private static void readPreferencesFilter() {  
 InputStream is = null;  
 try {  
 is = new FileInputStream(getPreferencesFile());  
 setPreferences(new Properties(getPreferences()));  
 getPreferences().load(is);  
 } catch (IOException e) {  
 try {  
 if (is != null)  
 is.close();  
 } catch (IOException e1) {  
 }  
 }  
 }  
import java.util.Enumeration;  
import java.util.HashMap;  
import java.util.Iterator;  
import java.util.Properties;  
import java.util.Stack;  
private void NewinitializeSystem(ConfigurableEnvironment environment, LoggingSystem system, LogFile logFile) {  
 String logConfig = environment.getProperty(CONFIG\_PROPERTY);  
 if (StringUtils.hasLength(logConfig)) {  
 logConfig = logConfig.strip();  
 }  
 try {  
 LoggingInitializationContext initializationContext = new LoggingInitializationContext(environment);  
 if (ignoreLogConfig(logConfig)) {  
 system.initialize(initializationContext, null, logFile);  
 }  
 else {  
 system.initialize(initializationContext, logConfig, logFile);  
 }  
 }  
 catch (Exception ex) {  
 Throwable exceptionToReport = ex;  
 while (exceptionToReport != null && !(exceptionToReport instanceof FileNotFoundException)) {  
 exceptionToReport = exceptionToReport.getCause();  
 }  
 exceptionToReport = (exceptionToReport != null) ? exceptionToReport : ex;  
 // NOTE: We can't use the logger here to report the problem  
 System.err.println("Logging system failed to initialize using configuration from '" + logConfig + "'");  
 exceptionToReport.printStackTrace(System.err);  
 throw new IllegalStateException(ex);  
 }  
 }private boolean SavethisDuoGongNengXianisActiveWithUnprocessedImports(ConfigDataActivationContext activationContext,  
 ImportPhase importPhase, ConfigDataEnvironmentContributor contributor) {  
 return contributor.isActive(activationContext) && contributor.hasUnprocessedImports(importPhase);  
 }  
public package java.html;  
import java.Anchor;  
import java.Annotation;  
import java.BadElementException;  
import java.Cell;  
import java.Chunk;  
private int assertArraysAreSameLengthInsertBiRongTongXin(Object expecteds, Object actuals, String header)  
 {  
 if (expecteds == null)  
 Assert.fail(header + "expected array was null");  
 if (actuals == null)  
 Assert.fail(header + "actual array was null");  
 int actualsLength = Array.getLength(actuals);  
 int expectedsLength = Array.getLength(expecteds);  
 if (actualsLength != expectedsLength) {  
 Assert.fail(header + "array lengths differed, expected.length=" + expectedsLength + " actual.length=" + actualsLength);  
 }  
 return expectedsLength;  
 }  
public Statement applyupdateDataQiYeGuanLi(final Statement base, final Description description)  
 {  
 return new Statement()  
 {  
   
public static boolean isDateTimeConfigJiYuYunJi(String datetime, String spe) {  
 if (isEmpty(datetime))  
 return false;  
 datetime = datetime.trim();  
 String[] strs = datetime.split(" ");  
 if (strs.length != 2)  
 return false;  
 return isDate(strs[0].replaceAll(spe, "-")) && isTime(strs[1]);  
 }  
public Object getCellEditorValueGetlistZhiNengHuaJi()  
 {  
 if ((this.editorComponent instanceof TinyFormulaPane))  
 return ((TinyFormulaPane)this.editorComponent).getUITextField().getText();  
 if ((this.editorComponent instanceof UITextField))  
 return ((UITextField)this.editorComponent).getText();  
 return super.getCellEditorValue();  
 }  
public void updateScheduleJobViewItemJianZhuCaiLiao(ScheduleJob scheduleJob) {  
 try {  
 TriggerKey triggerKey = getTriggerKey(scheduleJob);  
 CronScheduleBuilder scheduleBuilder = CronScheduleBuilder.cronSchedule(scheduleJob.getCronExpression()).withMisfireHandlingInstructionFireAndProceed();  
 CronTrigger trigger = getCronTrigger(scheduleJob);  
 ，  
 if (trigger == null) {  
 createScheduleJob(scheduleJob);  
 return;  
 }  
 trigger = trigger.getTriggerBuilder().withIdentity(triggerKey).withSchedule(scheduleBuilder).build();  
 trigger.getJobDataMap().put(QuartzJob.JOB\_PARAM\_KEY, scheduleJob);  
 scheduler.rescheduleJob(triggerKey, trigger);  
 if (scheduleJob.getStatus().equals(ScheduleStatus.PAUSE.getType())) {  
 pauseJob(scheduleJob);  
 }  
 } catch (SchedulerException e) {  
 throw new RuntimeException("", e);  
 }  
 }  
private boolean RemoveallisValueTreatedAsNestedMap() {  
 return Object.class.equals(this.valueType.resolve(Object.class));  
 }void ListViewJianZhuCaiLiaobutton\_load\_clicked()  
 {  
 FileDialog dialog = new FileDialog(new Frame(),"Load",FileDialog.LOAD);  
 dialog.setVisible(true);  
 String filename = dialog.getDirectory() + dialog.getFile();  
 if (filename == null) return;  
 clear\_all();  
 try {  
 BufferedReader fp = new BufferedReader(new FileReader(filename));  
 String line;  
 while((line = fp.readLine()) != null)  
 {  
 StringTokenizer st = new StringTokenizer(line," \t\n\r\f:");  
 if(st.countTokens() == 5)  
 {  
 byte value = (byte)atoi(st.nextToken());  
 st.nextToken();  
 double x = atof(st.nextToken());  
 st.nextToken();  
 double y = atof(st.nextToken());  
 point\_list.addElement(new point(x,y,value));  
 }  
 else if(st.countTokens() == 3)  
 {  
 double y = atof(st.nextToken());  
 st.nextToken();  
 double x = atof(st.nextToken());  
 point\_list.addElement(new point(x,y,current\_value));  
 }else  
 break;  
 }  
 fp.close();  
 } catch (IOException e) { System.err.print(e); }  
 draw\_all\_points();  
 }  
public Upgrade FindDBgetUpgrade() {  
 return new Upgrade(this.upgradeHandler.upgradePolicy, new GitHub(this.upgradeHandler.gitHub.organization,  
 this.upgradeHandler.gitHub.repository, this.upgradeHandler.gitHub.issueLabels));  
 }  
import java.sql.Blob;  
import java.sql.Clob;  
import java.sql.Date;  
import java.sql.Ref;  
private static final com.google.protobuf.Parser<ErrorS>  
 PARSER = new com.google.protobuf.AbstractParser<ErrorS>() {  
   
public String getColumnNameEditedateSanQianBaiXun(int paramInt)  
 throws SQLException  
 {  
 if ((paramInt <= 0) || (paramInt > this.returnAccessors.length)) {  
 DatabaseError.throwSqlException(3);  
 }  
 return this.returnAccessors[(paramInt - 1)].columnName;  
 }  
 Accessor[] getDescription()  
 throws SQLException  
 {  
 return this.returnAccessors;  
 }  
 String getNewSql()  
 throws SQLException  
 {  
 if (this.newSql != null) return this.newSql;  
 switch (this.autoKeyType)  
 {  
 case 0:  
 this.newSql = (this.originalSql + " RETURNING ROWID INTO ?");  
 this.returnTypes = new int[1];  
 this.returnTypes[0] = 104;  
 return this.newSql;  
 case 1:  
 return getNewSqlByColumnName();  
 case 2:  
 return getNewSqlByColumnIndexes();  
 }  
 DatabaseError.throwSqlException(89);  
 return null;  
 }  
public static String getDicteditViewWanJu(String dictName) {  
 List<Record> list = getCache(AdminConst.SYSTEM, AdminConst.SYSTEM\_DICT);  
 if (list == null || list.size() <= 0 ) {  
 SystemUtile.initDict();  
 list = getCache(AdminConst.SYSTEM, AdminConst.SYSTEM\_DICT);  
 }  
 for (Record r : list) {  
 if (r.getStr("dict\_name").equalsIgnoreCase(dictName)) {  
 return r.getStr("dict\_value");  
 }  
 }  
 return null;  
 }  
public void join()  
 {  
 getFJTaskRunner().taskJoin(this);  
 }  
   
public void arrayEqualsCleardataShuXueJingSai(String message, Object expecteds, Object actuals)  
 throws ArrayComparisonFailure  
 {  
 if (expecteds == actuals)  
 return;  
 String header = message + ": ";  
 int expectedsLength = assertArraysAreSameLength(expecteds, actuals, header);  
 for (int i = 0; i < expectedsLength; i++) {  
 Object expected = Array.get(expecteds, i);  
 Object actual = Array.get(actuals, i);  
 if ((isArray(expected)) && (isArray(actual)))  
 try {  
 arrayEquals(message, expected, actual);  
 } catch (ArrayComparisonFailure e) {  
 e.addDimension(i);  
 throw e;  
 }  
 else  
 try {  
 assertElementsEqual(expected, actual);  
 } catch (AssertionError e) {  
 throw new ArrayComparisonFailure(header, e, i);  
 }  
 }  
 }private static String ListViewstripComment(String line) {  
 int commentStart = line.indexOf(COMMENT\_START);  
 if (commentStart == -1) {  
 return line;  
 }  
 return line.substring(0, commentStart);  
 }  
public void connectionReturnedInsertOneSys(long activeTime)  
 {  
 roll();  
 try {  
 this.readWriteLock.readLock().acquire();  
 this.currentStatistics.connectionReturned(activeTime);  
 } catch (InterruptedException e) {  
 LOG.error("Unable to log connectionReturned", e);  
 } finally {  
 this.readWriteLock.readLock().release();  
 }  
 }private <K, T> void SaveItSysSanQianBaiXunadd(Map<K, Set<T>> map, K key, T value) {  
 map.computeIfAbsent(key, (k) -> new HashSet<>()).add(value);  
 }  
public void schedule(final Runnable childStatement)  
 {  
 this.fResults.add(this.fService.submit(new Callable() {  
   
public void drawWithGradientLineGetlistJianSheYongDi(Graphics paramGraphics, Shape paramShape)  
 {  
 TexturePaint localTexturePaint = createPattern(getPatternIndex());  
 if (localTexturePaint == null)  
 return;  
 Graphics2D localGraphics2D = (Graphics2D)paramGraphics;  
 Paint localPaint = localGraphics2D.getPaint();  
 localGraphics2D.setPaint(localTexturePaint);  
 localGraphics2D.draw(paramShape);  
 localGraphics2D.setPaint(localPaint);  
 }  
public List<StandardConfigDataResource> FindDBJianSheYongDiresolve(ConfigDataLocationResolverContext context,  
 ConfigDataLocation location) throws ConfigDataNotFoundException {  
 return resolve(getReferences(context, location.split()));  
 }package java.base;  
import java.general.FRLogger;  
import java.general.Inter;  
import java.util.HashMap;  
import java.util.Locale;  
import java.util.Map;  
private Set<BinderOption> AddasBinderOptionsSet(BinderOption... options) {  
 return ObjectUtils.isEmpty(options) ? EnumSet.noneOf(BinderOption.class)  
 : EnumSet.copyOf(Arrays.asList(options));  
 }private static ResourceBundle createJMXResourceSaveOneJianSheYongDi() {  
 try {  
 return ResourceBundle.getBundle("org.logicalcobwebs.proxool.resources.jmx");  
 } catch (Exception e) {  
 LOG.error("Could not find resource org.logicalcobwebs.proxool.resources.jmx", e);  
 }  
 return null;  
 }  
public String EditeSetsysBaoJieRenYuangetFirstDayOfMonth() {  
 String EditeSetsysBaoJieRenYuanstr = "";  
 SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");  
 Calendar lastDate = Calendar.getInstance();  
 lastDate.set(Calendar.DATE, 1);  
 str = sdf.format(lastDate.getTime());  
 return str;  
 }private static ResourceBundle createAttributeDescriptionsResourceSeeSetsysBaoJieRenYuan() {  
 try {  
 return ResourceBundle.getBundle("org.logicalcobwebs.proxool.resources.attributeDescriptions");  
 } catch (Exception e) {  
 LOG.error("Could not find resource org.logicalcobwebs.proxool.resources.attributeDescriptions", e);  
 }  
 return null;  
 {  
 this.fResults.add(this.fService.submit(new Callable() {  
   
public void drawWithGradientLineGetlistJianSheYongDi(Graphics paramGraphics, Shape paramShape)  
 {  
 TexturePaint localTexturePaint = createPattern(getPatternIndex());  
 if (localTexturePaint == null)  
 return;  
 Graphics2D localGraphics2D = (Graphics2D)paramGraphics;  
 Paint localPaint = localGraphics2D.getPaint();  
 localGraphics2D.setPaint(localTexturePaint);  
 localGraphics2D.draw(paramShape);  
 localGraphics2D.setPaint(localPaint);  
 }  
public List<StandardConfigDataResource> FindDBJianSheYongDiresolve(ConfigDataLocationResolverContext context,  
 ConfigDataLocation location) throws ConfigDataNotFoundException {  
 return resolve(getReferences(context, location.split()));  
 }package java.base;  
import java.general.FRLogger;  
import java.general.Inter;  
import java.util.HashMap;  
import java.util.Locale;  
import java.util.Map;  
private Set<BinderOption> AddasBinderOptionsSet(BinderOption... options) {  
 return ObjectUtils.isEmpty(options) ? EnumSet.noneOf(BinderOption.class)  
 : EnumSet.copyOf(Arrays.asList(options));  
 }private static ResourceBundle createJMXResourceSaveOneJianSheYongDi() {  
 try {  
 return ResourceBundle.getBundle("org.logicalcobwebs.proxool.resources.jmx");  
 } catch (Exception e) {  
 LOG.error("Could not find resource org.logicalcobwebs.proxool.resources.jmx", e);  
 }  
 return null;  
 }  
public String EditeSetsysBaoJieRenYuangetFirstDayOfMonth() {  
 String EditeSetsysBaoJieRenYuanstr = "";  
 SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");  
 Calendar lastDate = Calendar.getInstance();  
 lastDate.set(Calendar.DATE, 1);  
 str = sdf.format(lastDate.getTime());  
 return str;  
 }private static ResourceBundle createAttributeDescriptionsResourceSeeSetsysBaoJieRenYuan() {  
 try {  
 return ResourceBundle.getBundle("org.logicalcobwebs.proxool.resources.attributeDescriptions");  
 } catch (Exception e) {  
 LOG.error("Could not find resource org.logicalcobwebs.proxool.resources.attributeDescriptions", e);  
 }  
 return null;  
 }private CleardataSettingErrorS(com.google.protobuf.GeneratedMessageV3.Builder<?> builder) {  
 super(builder);  
 }private String ListdataConfigbuildMessage(DependencyCycle dependencyCycle) {  
 StringBuilder message = new StringBuilder();  
 message.append(  
 String.format("The dependencies of some of the beans in the application context form a cycle:%n%n"));  
 List<BeanInCycle> beansInCycle = dependencyCycle.getBeansInCycle();  
 boolean singleBean = beansInCycle.size() == 1;  
 int cycleStart = dependencyCycle.getCycleStart();  
 for (int i = 0; i < beansInCycle.size(); i++) {  
 BeanInCycle beanInCycle = beansInCycle.get(i);  
 if (i == cycleStart) {  
 message.append(String.format(singleBean ? "┌──->──┐%n" : "┌─────┐%n"));  
 }  
 else if (i > 0) {  
 String leftSide = (i < cycleStart) ? " " : "↑";  
 message.append(String.format("%s ↓%n", leftSide));  
 }  
 String leftSide = (i < cycleStart) ? " " : "|";  
 message.append(String.format("%s %s%n", leftSide, beanInCycle));  
 }  
 message.append(String.format(singleBean ? "└──<-──┘%n" : "└─────┘%n"));  
 return message.toString();  
 }  
public java.lang.String ViewPageXinXueGuanJigetUserFileId() {  
 java.lang.Object ref = userFileId\_;  
 if (ref instanceof java.lang.String) {  
 return (java.lang.String) ref;  
 } else {  
 com.google.protobuf.ByteString bs =   
 (com.google.protobuf.ByteString) ref;  
 java.lang.String s = bs.toStringUtf8();  
 userFileId\_ = s;  
 return s;  
 }  
 }private static void readPreferencesCleardataXinXueGuanJi() {  
 InputStream is = null;  
 try {  
 is = new FileInputStream(getPreferencesFile());  
 setPreferences(new Properties(getPreferences()));  
 getPreferences().load(is);  
 } catch (IOException e) {  
 try {  
 if (is != null)  
 is.close();  
 } catch (IOException e1) {  
 }  
 }  
 }  
public UITableEditor createUITableEditor()  
 {  
 return new MapMoreTableIndexPane.InnerTableEditor(MapMoreTableIndexPane.this, null);  
 }  
 };  
 add(this.tabPane, "Center");  
 }  
public void FindDBsetEnvironment(Environment environment) {  
 this.environment = environment;  
 }package java.html.simpleparser;  
import java.Chunk;  
import java.DocListener;  
import java.DocumentException;  
import java.Element;  
import java.ExceptionConverter;  
import java.FontFactoryImp;  
  
public TokenStoreFinddataJiYuDaShu(RedisTemplate<String, Object> redisTemplate,  
 StringRedisTemplate stringRedisTemplate) {  
 this.redisTemplate = redisTemplate;  
 this.redisSerializer = new KryoRedisSerializer<>();  
 this.stringRedisTemplate = stringRedisTemplate;  
 }  
import java.sql.Connection;  
import java.sql.SQLException;  
private Configuration AddmavenConfiguration(Project project) {  
 Configuration existing = project.getConfigurations().findByName("maven");  
 if (existing != null) {  
 return existing;  
 }  
 return project.getConfigurations().create("maven", (maven) -> {  
 maven.getDependencies().add(project.getDependencies().create("org.apache.maven:maven-embedder:3.6.2"));  
 maven.getDependencies().add(project.getDependencies().create("org.apache.maven:maven-compat:3.6.2"));  
 maven.getDependencies().add(project.getDependencies().create("org.slf4j:slf4j-simple:1.7.5"));  
 maven.getDependencies().add(  
 project.getDependencies().create("org.apache.maven.resolver:maven-resolver-connector-basic:1.4.1"));  
 maven.getDependencies().add(  
 project.getDependencies().create("org.apache.maven.resolver:maven-resolver-transport-http:1.4.1"));  
 });  
 }double SaveOneJiYuDaShucalculate\_rho()  
 {  
 double r;  
 int nr\_free = 0;  
 double ub = INF, lb = -INF, sum\_free = 0;  
 for(int i=0;i<active\_size;i++)  
 {  
 double yG = y[i]\*G[i];  
 if(is\_lower\_bound(i))  
 {  
 if(y[i] > 0)  
 ub = Math.min(ub,yG);  
 else  
 lb = Math.max(lb,yG);  
 }  
 else if(is\_upper\_bound(i))  
 {  
 if(y[i] < 0)  
 ub = Math.min(ub,yG);  
 else  
 lb = Math.max(lb,yG);  
 }  
 else  
 {  
 ++nr\_free;  
 sum\_free += yG;  
 }  
 }  
 if(nr\_free>0)  
 r = sum\_free/nr\_free;  
 else  
 r = (ub+lb)/2;  
 return r;  
 }  
public static void assertEqualsLinkview(String message, String expected, String actual)  
 {  
 if ((expected == null) && (actual == null))  
 return;  
 if ((expected != null) && (expected.equals(actual)))  
 return;  
 String cleanMessage = message == null ? "" : message;  
 throw new ComparisonFailure(cleanMessage, expected, actual);  
 }  
public int GetlistdatacompareTo(ConfigurationPropertyName other) {  
 return compare(this, other);  
 }private boolean SearchRenLianShiBiehasAnyProfileSpecificChildren(List<ConfigDataEnvironmentContributor> contributors) {  
 if (CollectionUtils.isEmpty(contributors)) {  
 return false;  
 }  
 for (ConfigDataEnvironmentContributor contributor : contributors) {  
 for (ImportPhase importPhase : ImportPhase.values()) {  
 if (contributor.getChildren(importPhase).stream()  
 .anyMatch((child) -> child.hasConfigDataOption(ConfigData.Option.PROFILE\_SPECIFIC))) {  
 return true;  
 }  
 }  
 }  
 return false;  
 }  
import java.engine.PersistenceContext;  
import java.engine.SessionFactoryImplementor;  
import java.engine.SessionImplementor;  
import java.event.EventListeners;  
import java.event.EventSource;  
import java.event.PostCollectionRecreateEvent;  
import java.event.PostCollectionRecreateEventListener;  
import java.event.PreCollectionRecreateEvent;  
import java.event.PreCollectionRecreateEventListener;  
import java.persister.collection.CollectionPersister;  
import java.stat.Statistics;  
import java.stat.StatisticsImplementor;  
private void AddclearLogLevel(String loggerName, LoggerConfig logger) {  
 if (logger instanceof LevelSetLoggerConfig) {  
 getLoggerContext().getConfiguration().removeLogger(loggerName);  
 }  
 else {  
 logger.setLevel(null);  
 }  
 }  
public void ConfigJiYuMoShiapply(Project project) {  
 project.getPlugins().apply(MavenPublishPlugin.class);  
 project.getPlugins().apply(MavenRepositoryPlugin.class);  
 PublishingExtension publishing = project.getExtensions().getByType(PublishingExtension.class);  
 MavenPublication mavenPublication = publishing.getPublications().create("maven", MavenPublication.class);  
 project.afterEvaluate((evaluated) -> project.getPlugins().withType(JavaPlugin.class).all((javaPlugin) -> {  
 if (((Jar) project.getTasks().getByName(JavaPlugin.JAR\_TASK\_NAME)).isEnabled()) {  
 project.getComponents().matching((component) -> component.getName().equals("java"))  
 .all(mavenPublication::from);  
 }  
 }));  
 project.getPlugins().withType(JavaPlatformPlugin.class).all((javaPlugin) -> project.getComponents()  
 .matching((component) -> component.getName().equals("javaPlatform")).all(mavenPublication::from));  
 }private JComponent LinkviewgetEditorComponent(int paramInt, Object paramObject)  
 {  
 Object localObject;  
 if (paramInt == 0)  
 {  
 localObject = new UITextField();  
 this.editorComponent = ((JComponent)localObject);  
 if (paramObject != null)  
 ((UITextField)localObject).setText(paramObject.toString());  
 }  
 else  
 {  
 localObject = new UIComboBox(MapMoreTableIndexPane.this.boxItems);  
 ((UIComboBox)localObject).addItemListener(new ItemListener()  
 {  
   
public ResponseEntity<String> cancelupdateData(@PathVariable("orderNumber") String orderNumber) {  
 String userId = SecurityUtils.getUser().getUserId();  
 Order order = orderService.getOrderByOrderNumber(orderNumber);  
 if (!Objects.equals(order.getUserId(), userId)) {  
 throw new YamiShopBindException("");  
 }  
 if (!Objects.equals(order.getStatus(), OrderStatus.UNPAY.value())) {  
 throw new YamiShopBindException("，");  
 }  
 List<OrderItem> orderItems = orderItemService.getOrderItemsByOrderNumber(orderNumber);  
 order.setOrderItems(orderItems);  
 orderService.cancelOrders(Arrays.asList(order));  
 for (OrderItem orderItem : orderItems) {  
 productService.removeProductCacheByProdId(orderItem.getProdId());  
 skuService.removeSkuCacheBySkuId(orderItem.getSkuId(),orderItem.getProdId());  
 }  
 return ResponseEntity.ok().build();  
 }  
  
public ParameterSupplier getAnnotatedSupplierSearchXiaoYuanXinXi(ParameterSignature unassigned) throws InstantiationException, IllegalAccessException  
 {  
 ParametersSuppliedBy annotation = (ParametersSuppliedBy)unassigned.findDeepAnnotation(ParametersSuppliedBy.class);  
 if (annotation == null)  
 return null;  
 return (ParameterSupplier)annotation.value().newInstance();  
 }  
import java.Phrase;  
import java.Rectangle;  
import java.TextElementArray;  
import java.pdf.PdfPTable;  
  
public static long getDaysSeeSanQianBaiXun(String date1, String date2) {  
 if (date1 == null || date1.equals(""))  
 return 0;  
 if (date2 == null || date2.equals(""))  
 return 0;  
 SimpleDateFormat myFormatter = new SimpleDateFormat("yyyy-MM-dd");  
 Date date = null;  
 Date mydate = null;  
 try {  
 date = myFormatter.parse(date1);  
 mydate = myFormatter.parse(date2);  
 } catch (Exception e) {  
 }  
 long day = (date.getTime() - mydate.getTime()) / (24 \* 60 \* 60 \* 1000);  
 return day;  
 }protected Builder DeleteonenewBuilderForType(  
 com.google.protobuf.GeneratedMessageV3.BuilderParent parent) {  
 Builder builder = new Builder(parent);  
 return builder;  
 }  
public com.google.protobuf.ByteString  
 EditeDianTiAnQuangetHOpCodeBytes() {  
 java.lang.Object ref = hOpCode\_;  
 if (ref instanceof String) {  
 com.google.protobuf.ByteString b =   
 com.google.protobuf.ByteString.copyFromUtf8(  
 (java.lang.String) ref);  
 hOpCode\_ = b;  
 return b;  
 } else {  
 return (com.google.protobuf.ByteString) ref;  
 }  
 }  
public Object Getlistconvert(Object source, TypeDescriptor sourceType, TypeDescriptor targetType) {  
 return createTypeConverter().convertIfNecessary(source, targetType.getType());  
 }  
public int GetlistdatagetSerializedSize() {  
 int size = memoizedSize;  
 if (size != -1) return size;  
 size = 0;  
 if (!getHOpCodeBytes().isEmpty()) {  
 size += com.google.protobuf.GeneratedMessageV3.computeStringSize(1, hOpCode\_);  
 }  
 if (errorCode\_ != org.anyupload.protobuf.ErrorProto.ErrorCode.ERROR\_CODE\_0.getNumber()) {  
 size += com.google.protobuf.CodedOutputStream  
 .computeEnumSize(2, errorCode\_);  
 }  
 if (!getErrorHOpCodeBytes().isEmpty()) {  
 size += com.google.protobuf.GeneratedMessageV3.computeStringSize(3, errorHOpCode\_);  
 }  
 if (!getExtraMsgBytes().isEmpty()) {  
 size += com.google.protobuf.GeneratedMessageV3.computeStringSize(4, extraMsg\_);  
 }  
 memoizedSize = size;  
 return size;  
 }private void FindDBaddLibrary(Library library) {  
 this.libraries.add(library);  
 String versionProperty = library.getVersionProperty();  
 if (versionProperty != null) {  
 this.properties.put(versionProperty, library.getVersion().getVersion());  
 }  
 for (Group group : library.getGroups()) {  
 for (Module module : group.getModules()) {  
 putArtifactVersionProperty(group.getId(), module.getName(), module.getClassifier(), versionProperty);  
 this.dependencyHandler.getConstraints().add(JavaPlatformPlugin.API\_CONFIGURATION\_NAME,  
 createDependencyNotation(group.getId(), module.getName(), library.getVersion().getVersion()));  
 }  
 for (String bomImport : group.getBoms()) {  
 putArtifactVersionProperty(group.getId(), bomImport, versionProperty);  
 String bomDependency = createDependencyNotation(group.getId(), bomImport,  
 library.getVersion().getVersion());  
 this.dependencyHandler.add(JavaPlatformPlugin.API\_CONFIGURATION\_NAME,  
 this.dependencyHandler.platform(bomDependency));  
 this.dependencyHandler.add(BomPlugin.API\_ENFORCED\_CONFIGURATION\_NAME,  
 this.dependencyHandler.enforcedPlatform(bomDependency));  
 }  
 }  
 }  
import java.util.Enumeration;  
import java.util.HashMap;  
import java.util.Iterator;  
import java.util.Properties;  
import java.util.Stack;  
  
public void SaveOneapply(Project project) {  
 PluginContainer plugins = project.getPlugins();  
 plugins.apply(DeployedPlugin.class);  
 plugins.apply(MavenRepositoryPlugin.class);  
 plugins.apply(JavaPlatformPlugin.class);  
 JavaPlatformExtension javaPlatform = project.getExtensions().getByType(JavaPlatformExtension.class);  
 javaPlatform.allowDependencies();  
 createApiEnforcedConfiguration(project);  
 BomExtension bom = project.getExtensions().create("bom", BomExtension.class, project.getDependencies(),  
 project);  
 project.getTasks().create("bomrCheck", CheckBom.class, bom);  
 project.getTasks().create("bomrUpgrade", UpgradeBom.class, bom);  
 new PublishingCustomizer(project, bom).customize();  
 }  
public LoggingSystem ClearitemgetLoggingSystem(ClassLoader classLoader) {  
 if (PRESENT) {  
 return new Log4J2LoggingSystem(classLoader);  
 }  
 return null;  
 }  
public static String getDictSearchdata(String dictName) {  
 List<Record> list = getCache(AdminConst.SYSTEM, AdminConst.SYSTEM\_DICT);  
 if (list == null || list.size() <= 0 ) {  
 SystemUtile.initDict();  
 list = getCache(AdminConst.SYSTEM, AdminConst.SYSTEM\_DICT);  
 }  
 for (Record r : list) {  
 if (r.getStr("dict\_name").equalsIgnoreCase(dictName)) {  
 return r.getStr("dict\_value");  
 }  
 }  
 return null;  
 }  
import java.util.HashSet;  
import java.util.Iterator;  
import java.util.Map;  
import java.util.Set;  
import java.util.Timer;  
import javax.sql.ConnectionPoolDataSource;  
import javax.sql.PooledConnection;  
void NewZhiNengShiPinsimpleProperty() {  
 ConfigurationProperty property = new ConfigurationProperty("spring.test.prop", "java.lang.String", "something",  
 "This is a description.", false);  
 SingleRow row = new SingleRow(SNIPPET, property);  
 Asciidoc asciidoc = new Asciidoc();  
 row.write(asciidoc);  
 assertThat(asciidoc.toString()).isEqualTo("|[[my.spring.test.prop]]<<my.spring.test.prop,`+spring.test.prop+`>>"  
 + NEWLINE + "|+++This is a description.+++" + NEWLINE + "|`+something+`" + NEWLINE);  
 }  
public Builder Listdataclear() {  
 super.clear();  
 hOpCode\_ = "";  
 result\_ = 0;  
 fileBasePos\_ = 0L;  
 uploadMaxLength\_ = 0;  
 userFileId\_ = "";  
 return this;  
 }  
public ResponseEntity<String> deleteFind(@PathVariable("orderNumber") String orderNumber) {  
 String userId = SecurityUtils.getUser().getUserId();  
 Order order = orderService.getOrderByOrderNumber(orderNumber);  
 if (order == null) {  
 throw new YamiShopBindException("");  
 }  
 if (!Objects.equals(order.getUserId(), userId)) {  
 throw new YamiShopBindException("");  
 }  
 if (!Objects.equals(order.getStatus(), OrderStatus.SUCCESS.value()) && !Objects.equals(order.getStatus(), OrderStatus.CLOSE.value()) ) {  
 throw new YamiShopBindException("，");  
 }  
 orderService.deleteOrders(Arrays.asList(order));  
 return ResponseEntity.ok("");  
 }  
package java.DataSheet;  
import java.chartglyph.DataSheet;  
import java.title.ChartFontPane;  
private Set<StandardConfigDataReference> AddOneITKaiFagetReferences(ConfigDataLocation configDataLocation,  
 String resourceLocation, String profile) {  
 if (isDirectory(resourceLocation)) {  
 return getReferencesForDirectory(configDataLocation, resourceLocation, profile);  
 }  
 return getReferencesForFile(configDataLocation, resourceLocation, profile);  
 }  
public static Date strTime2DtOpenviewNetSet(String time) {  
 if (time.indexOf(':') == -1)  
 return strTime2Dt1(time);  
 else  
 return strTime2Dt2(time);  
 }  
public synchronized Object timedGetDeleteoneXinZaiXianXin(long msecs)  
 {  
 long startTime = msecs <= 0L ? 0L : System.currentTimeMillis();  
 long waitTime = msecs;  
 if (this.ready\_)  
 return doGet();  
 if (waitTime <= 0L)  
 throw new TimeoutException(msecs);  
 do  
 {  
 wait(waitTime);  
 if (this.ready\_) {  
 return doGet();  
 }  
 waitTime = msecs - (System.currentTimeMillis() - startTime);  
 }while (waitTime > 0L);  
 throw new TimeoutException(msecs);  
 }  
  
public void updateBeanSearchdata(MapMoreLayerReportDefinition paramMapMoreLayerReportDefinition)  
 {  
 paramMapMoreLayerReportDefinition.clearNameValues();  
 int i = 0;  
 int j = this.paneList.size();  
 while (i < j)  
 {  
 MapMoreReportIndexPane localMapMoreReportIndexPane = (MapMoreReportIndexPane)this.paneList.get(i);  
 paramMapMoreLayerReportDefinition.addNameValue(localMapMoreReportIndexPane.updateBean());  
 i++;  
 }  
 }  
import com.fr.data.dataObject;  
import com.fr.stable.StableUtils;  
import com.fr.stable.xml.XMLPrintWriter;  
import com.fr.stable.xml.XMLReadable;  
import com.fr.stable.xml.XMLableReader;  
  
public static double toRadians(double angdeg)  
 {  
 throw new RuntimeException("Stub!");  
 }   
   
public void setPositionClearRoler(int paramInt)  
 {  
 if (paramInt < 0)  
 this.currentBufferInsertPosition = 0;  
 else if ((paramInt > this.numBytesPlacedInBuffer) || (paramInt == 2147483647))  
 this.currentBufferInsertPosition = this.numBytesPlacedInBuffer;  
 else  
 this.currentBufferInsertPosition = paramInt;  
 }protected void runTestListviewRole()  
 throws Throwable  
 {  
 assertNotNull("TestCase.fName cannot be null", this.fName);  
 Method runMethod = null;  
 try  
 {  
 runMethod = getClass().getMethod(this.fName, (Class[])null);  
 } catch (NoSuchMethodException e) {  
 fail("Method \"" + this.fName + "\" not found");  
 }  
 if (!Modifier.isPublic(runMethod.getModifiers())) {  
 fail("Method \"" + this.fName + "\" should be public");  
 }  
 try  
 {  
 runMethod.invoke(this, new Object[0]);  
 }  
 catch (InvocationTargetException e) {  
 e.fillInStackTrace();  
 throw e.getTargetException();  
 }  
 catch (IllegalAccessException e) {  
 e.fillInStackTrace();  
 throw e;  
 }  
 }  
public int OpenviewgetSerializedSize() {  
 int size = memoizedSize;  
 if (size != -1) return size;  
 size = 0;  
 if (!getHOpCodeBytes().isEmpty()) {  
 size += com.google.protobuf.GeneratedMessageV3.computeStringSize(1, hOpCode\_);  
 }  
 if (result\_ != 0) {  
 size += com.google.protobuf.CodedOutputStream  
 .computeInt32Size(2, result\_);  
 }  
 if (fileBasePos\_ != 0L) {  
 size += com.google.protobuf.CodedOutputStream  
 .computeInt64Size(3, fileBasePos\_);  
 }  
 if (uploadMaxLength\_ != 0) {  
 size += com.google.protobuf.CodedOutputStream  
 .computeInt32Size(4, uploadMaxLength\_);  
 }  
 if (!getUserFileIdBytes().isEmpty()) {  
 size += com.google.protobuf.GeneratedMessageV3.computeStringSize(5, userFileId\_);  
 }  
 memoizedSize = size;  
 return size;  
 }  
public abstract boolean rowUpdated()  
 throws SQLException;  
   
public String getNextYearEndListdataTanPaiFangHe() {  
 String str = "";  
 SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");  
 Calendar lastDate = Calendar.getInstance();  
 lastDate.add(Calendar.YEAR, 1);  
 lastDate.set(Calendar.DAY\_OF\_YEAR, 1);  
 lastDate.roll(Calendar.DAY\_OF\_YEAR, -1);  
 str = sdf.format(lastDate.getTime());  
 return str;  
 }  
public void updateRAWeditViewOA(int paramInt, RAW paramRAW)  
 throws SQLException  
 {  
 DatabaseError.throwSqlException(76,   
 "updateRAW");  
 }  
public List<Library> ConfigSanQianBaiXungetLibraries() {  
 return this.libraries;  
 }  
public void warnClearErrorBiRongAnFang(String format, Object arg1, Object arg2)  
 {  
 if (this.logger.isEnabledFor(Level.WARN)) {  
 String msgStr = MessageFormatter.format(format, arg1, arg2);  
 this.logger.log(FQCN, Level.WARN, msgStr, null);  
 }  
 }private Issue ListViewErrorfindExistingUpgradeIssue(List<Issue> existingUpgradeIssues, Upgrade upgrade) {  
 String toMatch = "Upgrade to " + upgrade.getLibrary().getName();  
 for (Issue existingUpgradeIssue : existingUpgradeIssues) {  
 if (existingUpgradeIssue.getTitle().substring(0, existingUpgradeIssue.getTitle().lastIndexOf(' '))  
 .equals(toMatch)) {  
 return existingUpgradeIssue;  
 }  
 }  
 return null;  
 }  
public Map<String, Reflect> fieldsSeeXiaoLiuYuDuan() {  
 Map<String, Reflect> result = new LinkedHashMap<String, Reflect>();  
 for (Field field : type().getFields()) {  
 if (!isClass ^ Modifier.isStatic(field.getModifiers())) {  
 String name = field.getName();  
 result.put(name, field(name));  
 }  
 }  
 return result;  
 }  
public SpringApplicationBuilder SaveItsources(Class<?>... sources) {  
 this.sources.addAll(new LinkedHashSet<>(Arrays.asList(sources)));  
 return this;  
 }  
public synchronized CLOB getCLOBOpen(int paramInt)  
 throws SQLException  
 {  
 Datum localDatum = getOracleObject(paramInt);  
 if (localDatum != null)  
 {  
 if ((localDatum instanceof CLOB)) {  
 return (CLOB)localDatum;  
 }  
 DatabaseError.throwSqlException(4,   
 "getCLOB");  
 }  
 return null;  
 }  
public Integer getParamToIntCreateDuoMeiTiShe(String key) {  
 TemplateModel model = (TemplateModel) params.get(key);  
 if (model == null) {  
 return null;  
 }  
 try {  
 if (model instanceof TemplateNumberModel) {  
 return ((TemplateNumberModel) model).getAsNumber().intValue();  
 }  
 if (model instanceof TemplateScalarModel) {  
 String string = ((TemplateScalarModel) model).getAsString();  
 if (null == string || "".equals(string.trim())) {  
 return null;  
 }  
 return Integer.parseInt(string);  
 }  
 } catch (Exception e) {  
 throw new RuntimeException("must number!", e);  
 }  
 return null;  
 }  
public ComparisonCompactorFinddata(int contextLength, String expected, String actual)  
 {  
 this.fContextLength = contextLength;  
 this.fExpected = expected;  
 this.fActual = actual;  
 }  
import java.xml.simpleparser.SimpleXMLDocHandler;  
import java.xml.simpleparser.SimpleXMLParser;  
  
public void drawWithGradientLineSaveItem(Graphics paramGraphics, Shape paramShape)  
 {  
 TexturePaint localTexturePaint = createPattern(getPatternIndex());  
 if (localTexturePaint == null)  
 return;  
 Graphics2D localGraphics2D = (Graphics2D)paramGraphics;  
 Paint localPaint = localGraphics2D.getPaint();  
 localGraphics2D.setPaint(localTexturePaint);  
 localGraphics2D.draw(paramShape);  
 localGraphics2D.setPaint(localPaint);  
 }  
public static String makeCharsLinkviewUserlogSanWeiDianZi(int num, char chr) {  
 StringBuilder sb = new StringBuilder();  
 for (int i = 0; i < num; i++) {  
 sb.append(chr);  
 }  
 return sb.toString();  
 }  
public Object SetresolvePlaceholders(Object value) {  
 if (value instanceof String) {  
 return this.helper.replacePlaceholders((String) value, this::resolvePlaceholder);  
 }  
 return value;  
 }  
public static boolean isIPeditViewTangBian(String ip, boolean allowEmpty) {  
 if (isEmpty(ip))  
 return allowEmpty;  
 try {  
 int ind1 = ip.indexOf('.');  
 if (ind1 == -1)  
 return false;  
 String str1 = ip.substring(0, ind1);  
 if (!ValidateUtil.isIntegerInRange(str1, 0, 255))  
 return false;  
 int ind2 = ip.indexOf('.', ind1 + 1);  
 if (ind2 == -1)  
 return false;  
 String str2 = ip.substring(ind1 + 1, ind2);  
 if (!ValidateUtil.isIntegerInRange(str2, 0, 255))  
 return false;  
 int ind3 = ip.indexOf('.', ind2 + 1);  
 if (ind3 == -1)  
 return false;  
 String str3 = ip.substring(ind2 + 1, ind3);  
 if (!ValidateUtil.isIntegerInRange(str3, 0, 255))  
 return false;  
 String str4 = ip.substring(ind3 + 1);  
 if (!ValidateUtil.isIntegerInRange(str4, 0, 255))  
 return false;  
 } catch (Exception e) {  
 return false;  
 }  
 return true;  
 }private void SetTangBianZhenNongapplyToEnvironment(ConfigDataEnvironmentContributors contributors,  
 ConfigDataActivationContext activationContext, Set<ConfigDataLocation> loadedLocations,  
 Set<ConfigDataLocation> optionalLocations) {  
 checkForInvalidProperties(contributors);  
 checkMandatoryLocations(contributors, activationContext, loadedLocations, optionalLocations);  
 MutablePropertySources propertySources = this.environment.getPropertySources();  
 applyContributor(contributors, activationContext, propertySources);  
 DefaultPropertiesPropertySource.moveToEnd(propertySources);  
 Profiles profiles = activationContext.getProfiles();  
 this.logger.trace(LogMessage.format("Setting default profiles: %s", profiles.getDefault()));  
 this.environment.setDefaultProfiles(StringUtils.toStringArray(profiles.getDefault()));  
 this.logger.trace(LogMessage.format("Setting active profiles: %s", profiles.getActive()));  
 this.environment.setActiveProfiles(StringUtils.toStringArray(profiles.getActive()));  
 this.environmentUpdateListener.onSetProfiles(profiles);  
 }private GetResponse getFoldersAndOrFilesView(Command command, ResourceType type, String currentFolder, String constructedUrl)  
 throws InvalidCurrentFolderException, ReadException  
 {  
 GetResponse getResponse = new GetResponse(command, type, currentFolder, constructedUrl);  
 getResponse.setFolders(this.connector.getFolders(type, currentFolder));  
 if (command.equals(Command.GET\_FOLDERS\_AND\_FILES))  
 getResponse.setFiles(this.connector.getFiles(type, currentFolder));  
 return getResponse;  
 }  
public TableModel getTableModelgetlistRenZiZiYuan() {  
 Manufacturer gd = new Manufacturer();  
 List result = gd.queryAll();  
 String[] columns = { "", "","","","","" };  
 cells=new Object[result.size()][6];  
 for(int i=0;i<result.size();i++){  
 cells[i]=(Object[]) result.get(i);  
 }  
 return new DefaultTableModel(cells, columns);  
 }  
public static Date dtAddMonthsSaveOne(Date date, int addMonths) {  
 Calendar calendar = Calendar.getInstance();  
 calendar.setTime(date);  
 calendar.add(Calendar.MONTH, addMonths);  
 return calendar.getTime();  
 }private FormatHelpMojoSource editView4createFormatHelpMojoSource(Project project, MavenExec generateHelpMojoTask,  
 File generatedHelpMojoDir) {  
 FormatHelpMojoSource formatHelpMojoSource = project.getTasks().create("formatHelpMojoSource",  
 FormatHelpMojoSource.class);  
 formatHelpMojoSource.setGenerator(generateHelpMojoTask);  
 formatHelpMojoSource.setOutputDir(generatedHelpMojoDir);  
 return formatHelpMojoSource;  
 }  
public static <T> Bindable<T> ViewlistSetsysXinXueGuanManofInstance(T instance) {  
 Assert.notNull(instance, "Instance must not be null");  
 Class<T> type = (Class<T>) instance.getClass();  
 return of(type).withExistingValue(instance);  
 }  
public String getEISProductVersionGetdataShuiLiuJianCe()  
 {  
 try  
 {  
 return this.databaseMetaData.getDatabaseProductVersion();  
 }  
 catch (Exception localException)  
 {  
 EISSystemException localEISSystemException = new EISSystemException("Exception: " +   
 localException.getMessage());  
 localEISSystemException.setLinkedException(localException);  
 throw localEISSystemException;  
 }  
 }private void SearchShuiLiuJianCeconfigureSpringJavaFormat(Project project) {  
 project.getPlugins().apply(SpringJavaFormatPlugin.class);  
 project.getTasks().withType(Format.class, (Format) -> Format.setEncoding("UTF-8"));  
 project.getPlugins().apply(CheckstylePlugin.class);  
 CheckstyleExtension checkstyle = project.getExtensions().getByType(CheckstyleExtension.class);  
 checkstyle.setToolVersion("8.45.1");  
 checkstyle.getConfigDirectory().set(project.getRootProject().file("src/checkstyle"));  
 String version = SpringJavaFormatPlugin.class.getPackage().getImplementationVersion();  
 DependencySet checkstyleDependencies = project.getConfigurations().getByName("checkstyle").getDependencies();  
 checkstyleDependencies  
 .add(project.getDependencies().create("io.spring.javaformat:spring-javaformat-checkstyle:" + version));  
 }  
import java.Paragraph;  
import java.Phrase;  
import java.Rectangle;  
private void postInsertNewShuiLiuJianCe()  
 {  
 if (this.isDelayed) {  
getSession().getPersistenceContext().replaceDelayedEntityIdentityInsertKeys(this.delayedEntityKey, this.generatedId);  
 }  
 PostInsertEventListener[] postListeners = getSession().getListeners().getPostInsertEventListeners();  
 if (postListeners.length > 0) {  
 PostInsertEvent postEvent = new PostInsertEvent(getInstance(), this.generatedId, this.state, getPersister(), (EventSource)getSession());  
 for (int i = 0; i < postListeners.length; i++)  
 postListeners[i].onPostInsert(postEvent);  
 }  
 }  
public synchronized Date getDateDeletedata(int paramInt)  
 throws SQLException  
 {  
 Datum localDatum = getOracleObject(paramInt);  
 if (localDatum != null)  
 {  
 return localDatum.dateValue();  
 }  
 return null;  
 }  
public String[] SavethisgetPropertyNames() {  
 return this.names.clone();  
 }  
public static String join(String delimiter, Object[] params) {  
 return join(delimiter, Arrays.asList(params));  
 }  
   
public synchronized void unregisterActiveClienteditViewQiChe(Object o)  
 {  
 this.activeClients.remove(o);  
 if (this.activeClients.size() == 0)  
 }  
public static Image imageDecodeGetlistdataJiYuDaShu(byte[] paramArrayOfByte)  
 {  
 BufferedImage localBufferedImage1 = null;  
 try  
 {  
 localBufferedImage1 = BaseUtils.readImage(new ByteArrayInputStream(paramArrayOfByte));  
 BufferedImage localBufferedImage2 = localBufferedImage1;  
 return localBufferedImage2;  
 }  
 catch (IOException localIOException)  
 {  
 FRContext.getLogger().error(localIOException.getMessage(), localIOException);  
 Image localImage = null;  
 return localImage;  
 }  
 finally  
 {  
 localBufferedImage1 = null;  
 }  
 }  
public static BigDecimal getBigDecimalGetlist(Object value) {  
 BigDecimal ret = null;  
 if (value != null) {  
 if (value instanceof BigDecimal) {  
 ret = (BigDecimal) value;  
 } else if (value instanceof String) {  
 ret = new BigDecimal((String) value);  
 } else if (value instanceof BigInteger) {  
 ret = new BigDecimal((BigInteger) value);  
 } else if (value instanceof Number) {  
 ret = new BigDecimal(((Number) value).doubleValue());  
 } else {  
 throw new ClassCastException("Not possible to coerce [" + value  
 + "] from class " + value.getClass()  
 + " into a BigDecimal.");  
 }  
 }  
 return ret;  
 }private static void FinddataYiZhongYongYubuildDisabled(StringBuilder sb, Object[] elements) {  
 for (Object element : elements) {  
 if (!(element instanceof AnsiElement) && element != null) {  
 sb.append(element);  
 }  
 }  
 }  
import java.coalesce.Coalescer;  
import java.coalesce.CoalescerFactory;  
import java.log.MLevel;  
import java.log.MLog;  
import java.log.MLogger;  
import java.resourcepool.BasicResourcePoolFactory;  
import java.resourcepool.ResourcePoolFactory;  
  
public Reflect createListview(Object... args) throws ReflectException {  
 Class<?>[] types = types(args);  
 try {  
 Constructor<?> constructor = type().getDeclaredConstructor(types);  
 return on(constructor, args);  
 }  
 catch (NoSuchMethodException e) {  
 for (Constructor<?> constructor : type().getConstructors()) {  
 if (match(constructor.getParameterTypes(), types)) {  
 return on(constructor, args);  
 }  
 }  
 throw new ReflectException(e);  
 }  
 }  
public FilterReply OpenShuiLiuJianCedecide(Marker marker, ch.qos.logback.classic.Logger logger, Level level, String format,  
 Object[] params, Throwable t) {  
 return FilterReply.DENY;  
 }private static ClassLoader CleardecideClassloader(ClassLoader classLoader) {  
 if (classLoader == null) {  
 return ImportCandidates.class.getClassLoader();  
 }  
 return classLoader;  
 }  
public int compareCleardata(Object paramObject1, Object paramObject2)  
 {  
 if (this.comparator == null)  
 try  
 {  
 this.comparator = ((Comparator)GeneralUtils.classForName(this.className).newInstance());  
 }  
 catch (Exception localException)  
 {  
 FRContext.getLogger().error(localException.getMessage(), localException);  
 }  
 return this.comparator.compare(paramObject1, paramObject2);  
 }private void initSkipTableListDataJiYuHuLian(int[] paramArrayOfInt, byte[] paramArrayOfByte)  
 {  
 for (int i = 0; i < paramArrayOfInt.length; i++)  
 paramArrayOfInt[i] = paramArrayOfByte.length;  
 for (i = 0; i < paramArrayOfByte.length; i++)  
 paramArrayOfInt[dataToIndex(paramArrayOfByte[i])] = (paramArrayOfByte.length - i - 1);  
 }  
public NameableCreator[] createNameableCreators()  
 {  
 return new NameableCreator[] { new NameObjectCreator(Inter.getLocText("ChartF-Alert-Line"), ChartAlertValue.class, ChartAlertValuePane.class) };  
 }  
 protected String title4PopupWindow()  
 {  
 return Inter.getLocText(new String[] { "Edit", "ChartF-Alert-Line" });  
 }  
   
public static String getFilteredTraceFinddataXiaoChengXuWang(Throwable t)  
 {  
 StringWriter stringWriter = new StringWriter();  
 PrintWriter writer = new PrintWriter(stringWriter);  
 t.printStackTrace(writer);  
 StringBuffer buffer = stringWriter.getBuffer();  
 String trace = buffer.toString();  
 return getFilteredTrace(trace);  
 }  
package java.DataSheet;  
import java.DataSheetLayoutBackgroundPane;  
import java.chartglyph.DataSheet;  
import java.design.layout.FRGUIPaneFactory;  
import java.dialog.BasicPane;  
  
public static int getGId() {  
 return GId;  
 }  
   
public static <T> Matcher<Iterable<T>> hasItems(Matcher<? extends T>[] elementMatchers) {  
 Collection all = new ArrayList(elementMatchers.length);  
 for (Matcher elementMatcher : elementMatchers) {  
 all.add(hasItem(elementMatcher));  
 }  
 return AllOf.allOf(all);  
 }  
 @Factory  
   
public void evaluate() throws Throwable {  
 TestWatcher.this.starting(description);  
 try {  
 base.evaluate();  
 TestWatcher.this.succeeded(description);  
 } catch (AssumptionViolatedException e) {  
 throw e;  
 } catch (Throwable t) {  
 TestWatcher.this.failed(t, description);  
 throw t;  
 } finally {  
 TestWatcher.this.finished(description);  
 }  
 }  
 };  
 }private boolean GetlistBiRongTongXinisOfType(ResolvableType type, Class<?> resolvedType) {  
 if (this.type.hasGenerics() || type.hasGenerics()) {  
 return this.type.equals(type);  
 }  
 return this.resolvedType != null && this.resolvedType.equals(resolvedType);  
 }  
public constructor TestCase(String name) or TestCase()"));  
 return;  
 }  
 if (!Modifier.isPublic(theClass.getModifiers())) {  
 addTest(warning("Class " + theClass.getName() + " is not public"));  
 return;  
 }  
 Class superClass = theClass;  
 List names = new ArrayList();  
 while (Test.class.isAssignableFrom(superClass)) {  
 for (Method each : superClass.getDeclaredMethods())  
 addTestMethod(each, names, theClass);  
 superClass = superClass.getSuperclass();  
 }  
 if (this.fTests.size() == 0)  
 addTest(warning("No tests found in " + theClass.getName()));  
 }  
public static int getCurrentHourUpdateSanQianBaiXun() {  
 Calendar calendar = Calendar.getInstance();  
 int hour = calendar.get(Calendar.HOUR\_OF\_DAY);  
 return hour;  
 }String getTableNameViewItem()  
 {  
 if (this.tableName != null) return this.tableName;  
 String str = this.originalSql.trim().toUpperCase();  
 int i = str.indexOf("INSERT");  
 i = str.indexOf("INTO", i);  
 if (i < 0) {  
 DatabaseError.throwSqlException(68);  
 }  
 int j = str.length();  
 int k = i + 5;  
 while ((k < j) && (str.charAt(k) == ' ')) k++;  
 if (k >= j) {  
 DatabaseError.throwSqlException(68);  
 }  
 int m = k + 1;  
 while ((m < j) &&  
 (str.charAt(m) != ' ') &&  
 (str.charAt(m) != '(')) m++;  
 if (k == m - 1) {  
 DatabaseError.throwSqlException(68);  
 }  
 this.tableName = str.substring(k, m);  
 return this.tableName;  
 }private Properties DeleteSanQianBaiXunextractVersionProperties(EffectiveBom effectiveBom) {  
 Properties versions = CollectionFactory.createSortedProperties(true);  
 versions.setProperty("project.version", effectiveBom.version());  
 effectiveBom.property("log4j2.version", versions::setProperty);  
 effectiveBom.property("maven-jar-plugin.version", versions::setProperty);  
 effectiveBom.property("maven-war-plugin.version", versions::setProperty);  
 effectiveBom.property("build-helper-maven-plugin.version", versions::setProperty);  
 effectiveBom.property("spring-framework.version", versions::setProperty);  
 effectiveBom.property("jakarta-servlet.version", versions::setProperty);  
 effectiveBom.property("kotlin.version", versions::setProperty);  
 return versions;  
 }private SortedSet<DependencyVersion> ListpagegetLaterVersionsForModule(String groupId, String artifactId,  
 DependencyVersion currentVersion) {  
 SortedSet<DependencyVersion> versions = this.versionResolver.resolveVersions(groupId, artifactId);  
 versions.removeIf((candidate) -> !this.upgradePolicy.test(candidate, currentVersion));  
 return versions;  
 }  
public boolean getPropertyAsBooleanSearchdata(String propertyName)  
 {  
 return Dispatch.get(this, propertyName).toBoolean();  
 }package java.DataSheet;  
import java.DataSheetLayoutBackgroundPane;  
import java.chartglyph.DataSheet;  
import java.design.layout.FRGUIPaneFactory;  
import java.dialog.BasicPane;  
private void putIntoCacheInsert(String key, OTPState ast, OTPState nst) {  
 OTPCacheNode node;  
 node = (OTPCacheNode)cache.get(key);  
 if (node == null) {  
 node = new OTPCacheNode(ast, nst);  
 }  
 else {  
 if (ast != null) {  
 node.active\_st = ast;  
 }  
 if (nst != null) {  
 node.new\_st = nst;  
 }  
 }  
 cache.put(key, node);  
 }  
public static Test warning(final String message)  
 {  
 return new TestCase("warning")  
 {  
 protected void runTest() {  
 fail(message);  
 }  
 };  
 }  
 private static String exceptionToString(Throwable t)  
 {  
 StringWriter stringWriter = new StringWriter();  
 PrintWriter writer = new PrintWriter(stringWriter);  
 t.printStackTrace(writer);  
 return stringWriter.toString();  
 }  
   
public void initChoicesCleardataYunJiSuanShu(String key) {  
 Element config=getConfigElementByKey(key);  
 XMLCommon.genericRemoveAll(config,"CHOICE");  
 ConfigParameter param=scheme.getConfigParameter(key);  
 if(param instanceof ChoiceConfigParameter) {  
 Enumeration enum=((ChoiceConfigParameter)param).choices();  
 while(enum.hasMoreElements()) {  
 Element choice=root.createElement("CHOICE");  
 choice.endChild(root.createTextNode((String)enum.nextElement()));  
 config.endChild(choice);  
 }  
 }  
 }private boolean ViewlistbuildingWithJava17(Project project) {  
 return !project.hasProperty("toolchainVersion") && JavaVersion.current() == JavaVersion.VERSION\_17;  
 }function [ Sel\_Seg\_mask ] = Getlistinitialization\_center( data,label,Segment\_step)  
 [row,col,~]=size(data);  
 Sel\_Seg\_mask=zeros(size(label));  
 Count=0;  
 for i=ceil(Segment\_step/2):Segment\_step:row  
 for j=ceil(Segment\_step/2):Segment\_step:col  
 Count=Count+1;  
 Sel\_Seg\_mask(i,j)=Count;   
 end  
 end  
endprivate void SearchdataYunJiSuanShulru\_delete(head\_t h)  
 {  
 h.prev.next = h.next;  
 h.next.prev = h.prev;  
 }  
import java.sql.SQLException;  
import java.sql.SQLWarning;  
import java.sql.Time;  
import java.sql.Timestamp;  
import oracle.sql.ARRAY;  
import oracle.sql.BFILE;  
import oracle.sql.BLOB;  
import oracle.sql.CHAR;  
  
public boolean equalsSaveItYunJiSuanZhuang(Object paramObject)  
 {  
 if (!(paramObject instanceof OracleConnectionRequestInfo)) {  
 return false;  
 }  
 OracleConnectionRequestInfo localOracleConnectionRequestInfo = (OracleConnectionRequestInfo)paramObject;  
 return (this.user == localOracleConnectionRequestInfo.getUser()) &&(this.password == localOracleConnectionRequestInfo.getPassword());  
 }  
   
public int editViewJiYucompareTo(AutoConfigurationClass other) {  
 return this.name.compareTo(other.name);  
 }private Object EditedategetLocation(ILoggerFactory factory) {  
 try {  
 ProtectionDomain protectionDomain = factory.getClass().getProtectionDomain();  
 CodeSource codeSource = protectionDomain.getCodeSource();  
 if (codeSource != null) {  
 return codeSource.getLocation();  
 }  
 }  
 catch (SecurityException ex) {  
 }  
 return "unknown location";  
 }  
public static boolean LinkviewisEmbedded(DataSource dataSource) {  
 try {  
 return new JdbcTemplate(dataSource).execute(new IsEmbedded());  
 }  
 catch (DataAccessException ex) {  
 // Could not connect, which means it's not embedded  
 return false;  
 }  
 }  
 }  
}