

# Migrating to Neon

Judith Gull, Matthias Zimmermann



# Agenda

**What is Changing with Neon?**

**Migrating the «Contacts» Application**

→ Shared

→ Server

→ Client

→ Demo

**Migration Tips**

Migration to Neon  
**What is changing?**

# What changes in Neon?

HTML-UI  
Instead of RAP/Swing/SWT

Pure Java  
Instead of OSGi Bundles

Other API changes

# HTML-UI Changes

**Goals:** No conceptual changes in scout client model

➤ Possible to use existing models with minor modifications

**Migration:**

- Rewrite custom Swing/SWT/RAP components in JS/HTML
- New Attributes (CSS class)
- Fonts instead of Icons
- ...

# Vanilla Java

**Challenge:** Eclipse and OSGi code had to be removed

➤ No deprecations

## **Migration:**

- Convert project structure to maven default
- Convert manifest dependencies to maven dependencies
- Convert product files to launch files
- Use bean manager instead of OSGi service registry, no service registration
- Eclipse Jobs -> scout jobs
- Test runner

## Other API Changes

- Some renames/moves: AbstractExtensibleSmartField -> AbstractSmartField
- Improved Security: HTML is now escaped by default
- ClientNotification: Improved performance and API
- ...

# Migration Guide

- Not ready yet [1]
  - Currently not much experience with migrations (BSI Standard Product)
  - More projects expected to migrate in 2016

[1] <https://github.com/BSI-Business-Systems-Integration-AG/org.eclipse.scout.docs/blob/releases/5.2.x/docs/adoc/migration/MigrationGuide.adoc>



Migrating «Contacts»  
**A practical Example**

# «Contacts» Mars

The screenshot shows a web browser window with the URL `localhost:8082/web#org.eclipse.scout.contacts.client.ui.desktop.outlines.StandardOutline-Contacts`. The application is titled "Contacts" and is in "Event Management" mode. A sidebar on the left shows a tree view with "Contacts", "Company", and "Events". The main area displays a table of contacts:

First Name	Last Name	City	Country	Company	Events
Alice		Daresbury, Cheshire	GB	Alice's Adventures in Wonderland	2
Rabbit	the White	Daresbury, Cheshire	GB	Alice's Adventures in Wonderland	1

A "Contact" dialog box is open, showing details for the contact "Alice". It includes a small image of Alice and the White Rabbit. The form fields are:

- First Name: Alice
- Last Name: (empty)
- Date of Birth: Nov 26, 1865
- Gender:  Male  Female

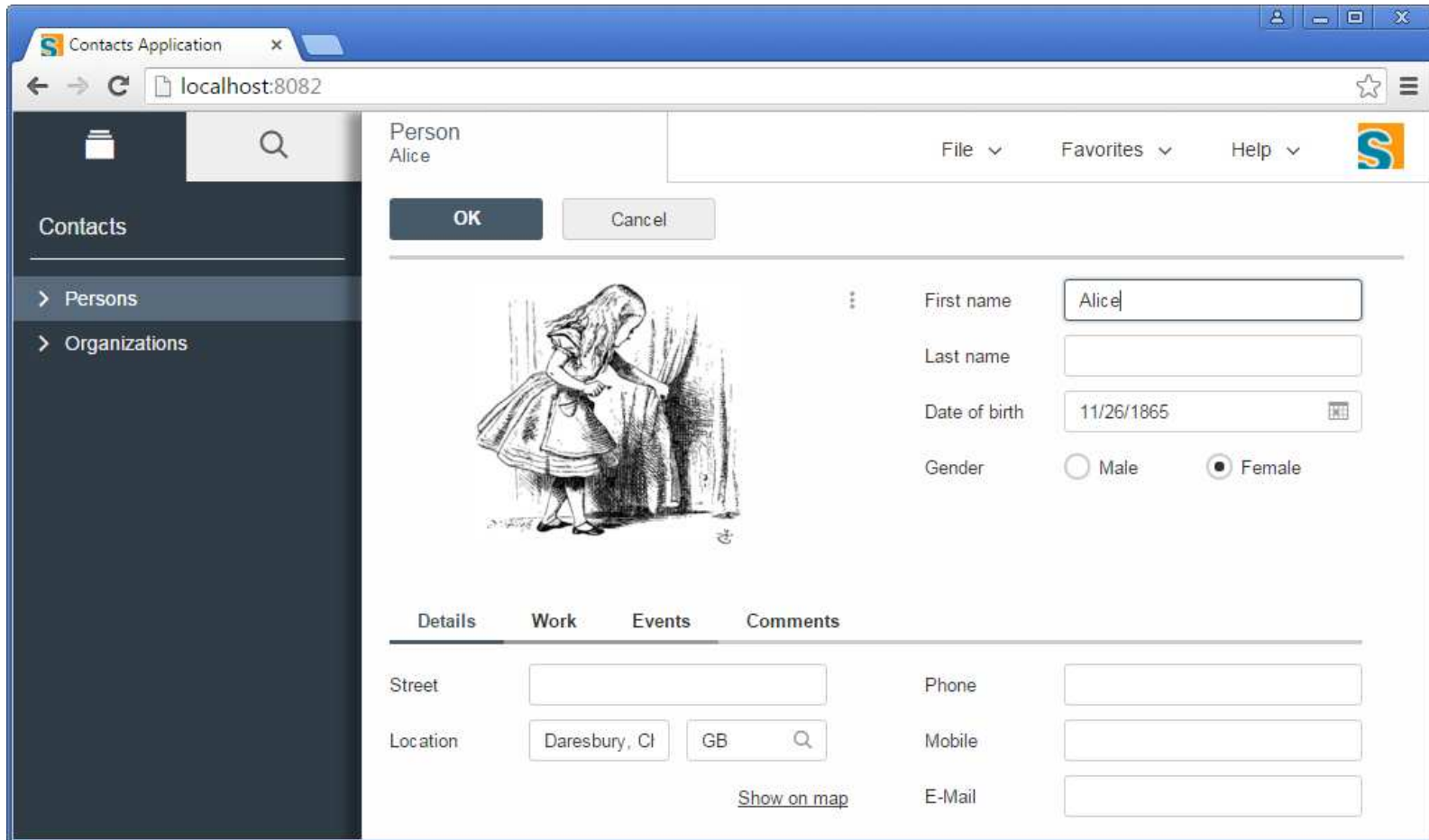
Below the form, there are tabs for "Details", "Work", "Events", and "Comments". The "Events" tab is active, showing a table of events:

Title	Starts	City	Country
★ EclipseCon 2015	3/9/15 9:00 AM	San Franc	US
★ JavaLand 2015	3/24/15 9:00 AM	Bruehl	DE

At the bottom of the dialog are "OK" and "Cancel" buttons. In the background, there are search fields for "City" and "Country" with "Reset" and "Search" buttons.

<https://github.com/BSI-Business-Systems-Integration-AG/org.eclipse.scout.docs/tree/releases/5.0.x/code/contacts>

# «Contacts» Neon



<https://github.com/BSI-Business-Systems-Integration-AG/org.eclipse.scout.docs/tree/releases/5.2.x/code/contacts>  
<https://github.com/BSI-Business-Systems-Integration-AG/org.eclipse.scout.docs/tree/releases/6.0.x/code/contacts>

# «Contacts»

## **Background**

- Replaces Wiki «Minicrm» and Scout Book «My Contacts»
- Broad coverage of Scout features
- Showcase release migrations

## **Neon «Business» Updates**

- Code organisation along business components
- Includes new features of HTML5 UI
  - Tabs replace modal forms
  - Tools
  - Theming

# Migrating the «Contacts»

## High level

1. Create a «Hello World» using the Scout SDK
2. Move existing business code to the new app
3. Remove errors and warnings

## Lower level

- ➔ Migrate the core, then the dependent business modules
- ➔ Migrate a module
  - First **shared**
  - Then **server**
  - Then **client**

Migrating «Contacts»  
Shared

# Migrating «Contacts»

## org.eclipsescout.contacts.shared

### Just copy & paste

- Permissions
- Codetypes & Codes
- Lookup calls
- FormData & PageData

### To modify for Neon

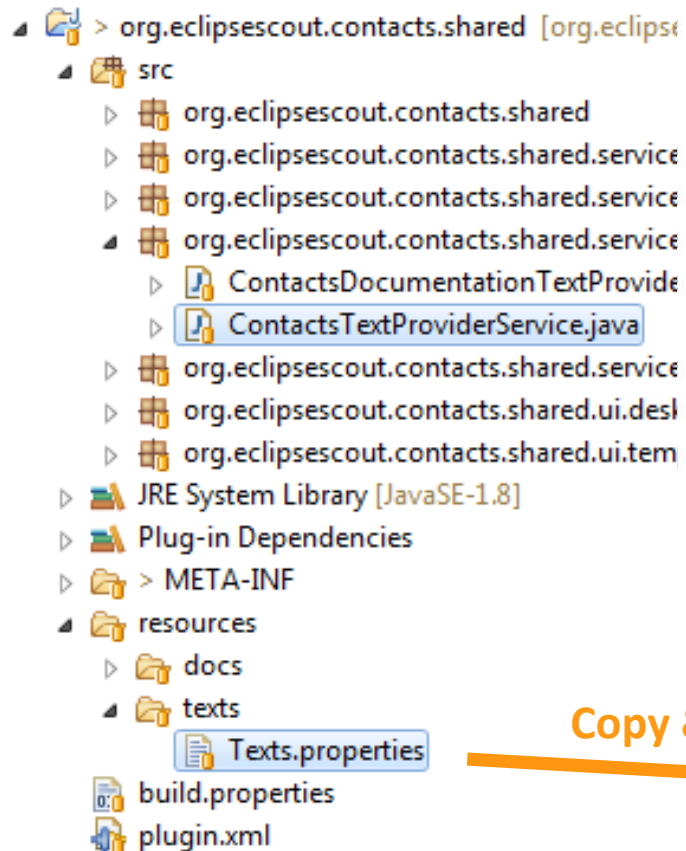
- Move `Texts.properties` to new place
- Lookup call interfaces
- Custom service interfaces

### Skipping

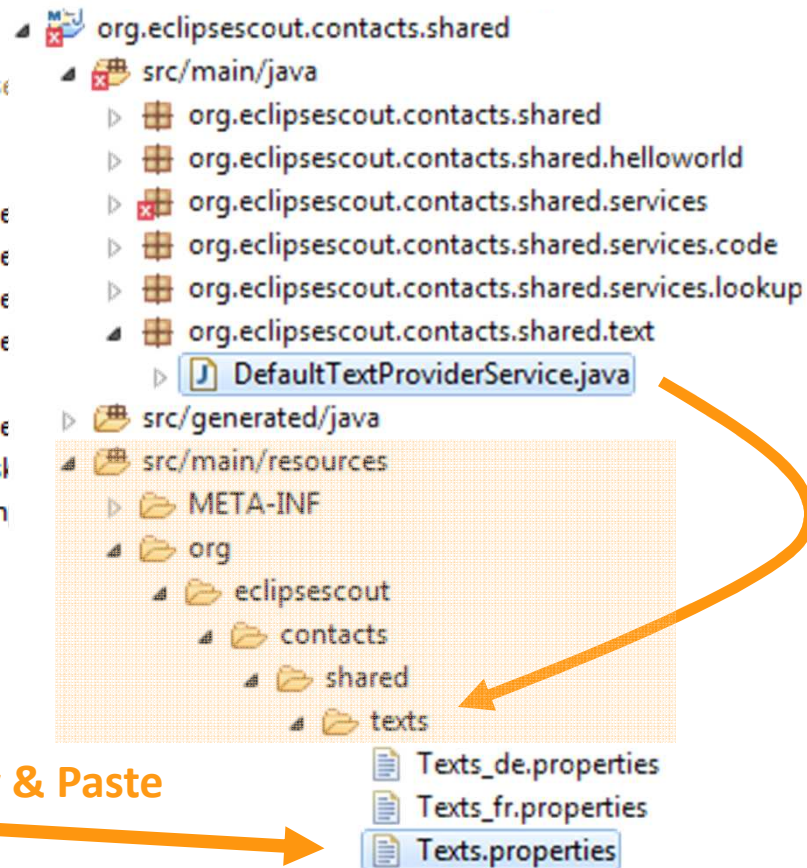
- `DefaultTextProviderService`

# Migrating «Contacts» Texts.properties

## Mars



## Neon



Copy & Paste



# Migrating «Contacts» Lookup Call Interfaces

## Mars

```
public interface IContactLookupService extends  
ILookupService<String> { }
```

## Neon

```
@TunnelToServer
```

```
public interface IContactLookupService extends  
ILookupService<String> { }
```

# Migrating «Contacts» Custom Service Interfaces

## Mars

```
@InputValidation( IValidationStrategy.PROCESS.class)
public interface IContactService extends IService {
    ...
}
```

## Neon

```
@ApplicationScoped
@TunnelToServer
@InputValidation( IValidationStrategy.PROCESS.class)
public interface IContactService {
    ...
}
```

# Migrating «Contacts» Server

# Migrating «Contacts»

## org.eclipsescout.contacts.server

### Just copy & paste

- Lookup services

### To modify for Neon

- Add **ConfigProperties**, **SuperUserRunContextProvider**, **PlatformListener**
- Move content of config.ini to **config.properties**
- Move startup logic to PlatformListener
- DerbySqlService (pom.xml ...)
- Custom services

### Skipping

- ServerSession
- AccessControlService

# Migrating «Contacts» Helper Classes (ConfigProperties ...)

## Neon

```
19 public final class ConfigProperties {
20
21     private ConfigProperties() {
22     }
23
24     public static class DatabaseJdbcMappingNameProperty extends AbstractStringConfigProperty {
25
26         @Override
27         protected String getDefaultValue() {
28             return "jdbc:derby:memory:contacts-database;create=true";
29         }
30
31         @Override
32         public String getKey() {
33             return "contacts.database.jdbcMappingName";
34         }
35     }
36 }
```

ServerServletFilter.java

ServerSession.java

SuperUserRunContextProvider.java

# Migrating «Contacts» Module per Product

## Mars

- org.eclipsescout.contacts.serv
  - src
  - JRE System Library [JavaSE
  - Plug-in Dependencies
  - j2ee
  - META-INF
  - products
    - development
      - config.ini
      - contacts-server-de
    - production

Adapt content

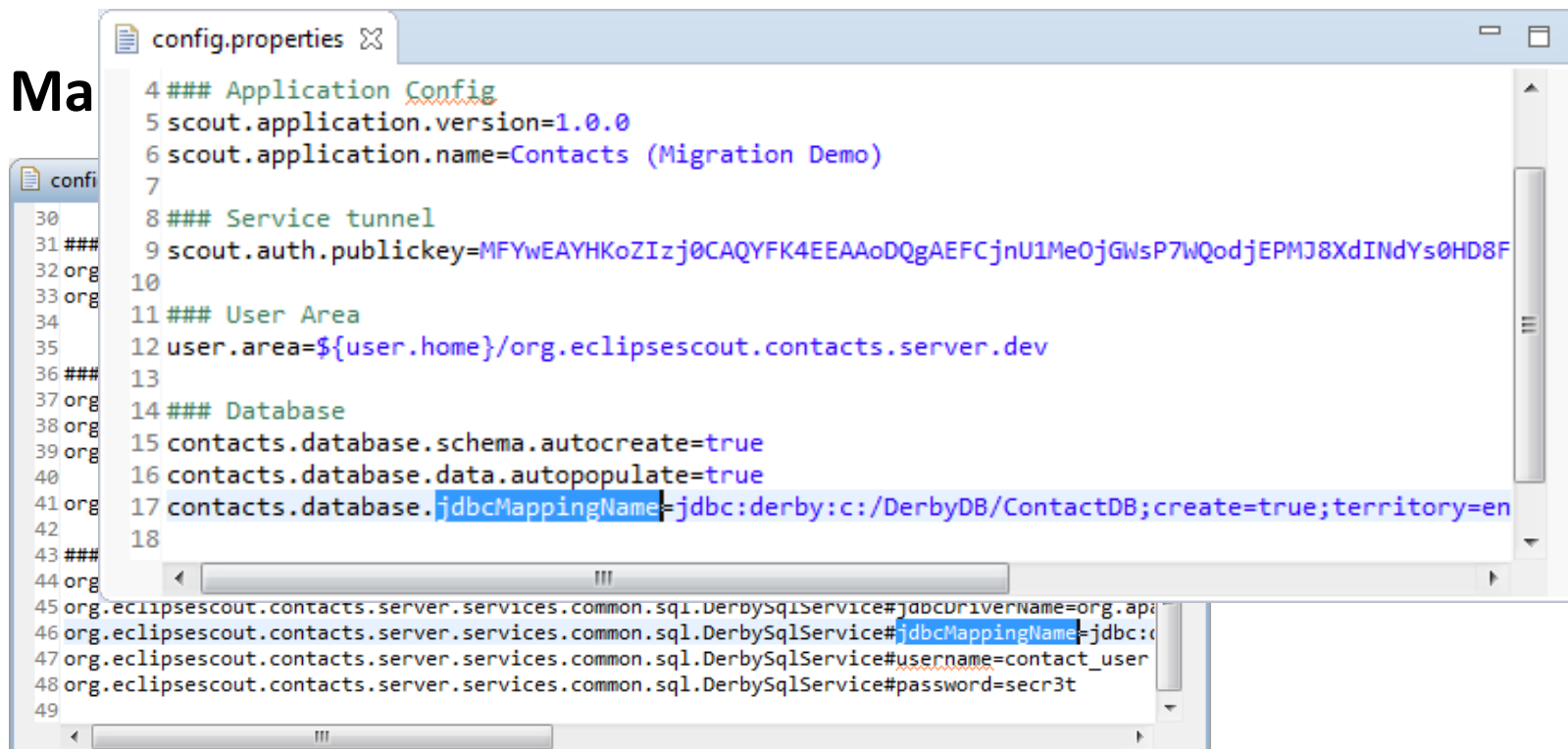
## Neon

- org.eclipsescout.contacts.server
- org.eclipsescout.contacts.server.app.dev
  - src/main/webapp
  - src/main/resources
    - config.properties
    - logback.xml
  - JRE System Library [JavaSE-1.8]
  - Maven Dependencies
  - src
  - target
  - pom.xml
  - server-dev.launch
- org.eclipsescout.contacts.server.app.war

# Migrating «Contacts» config.ini – config.properties

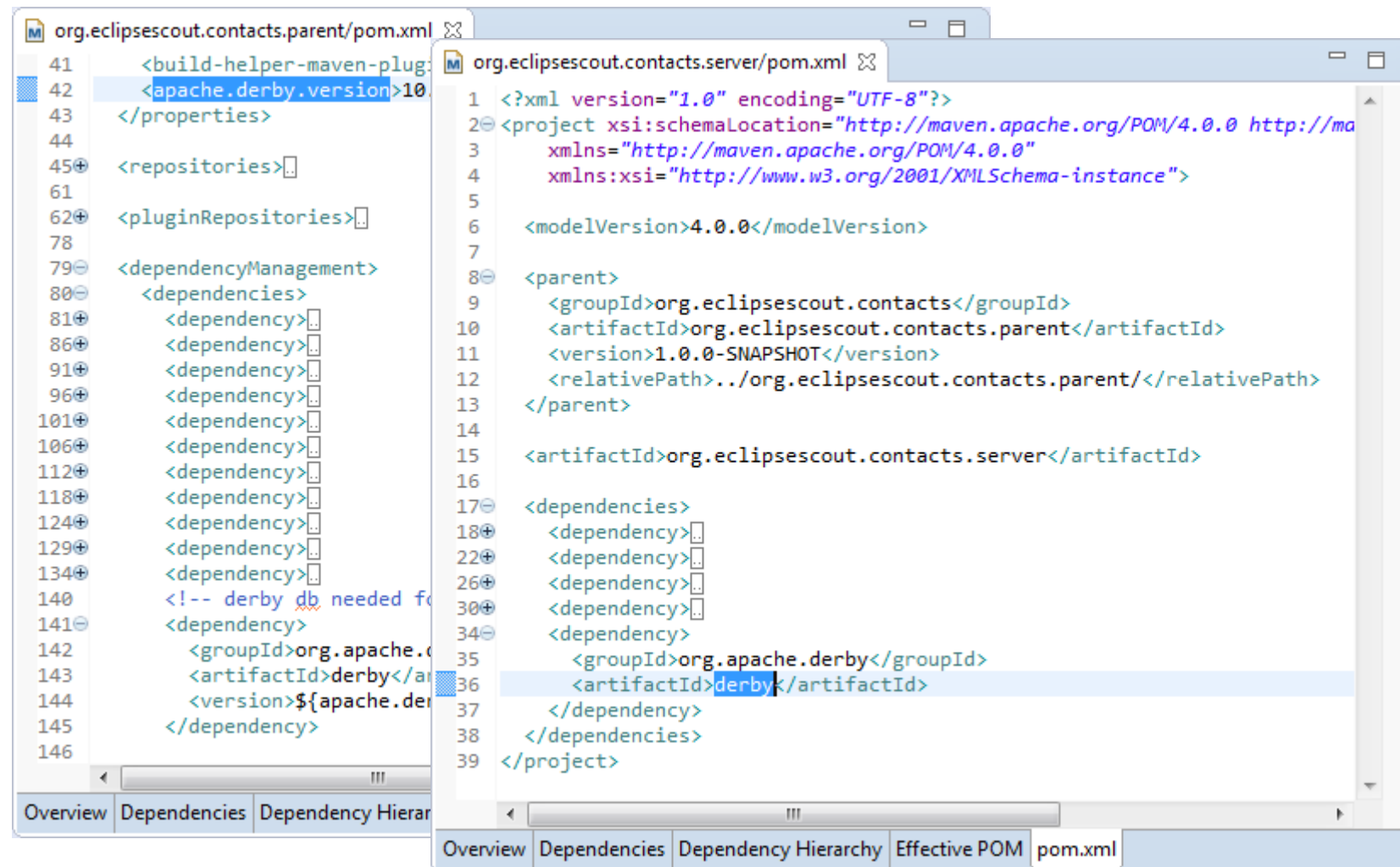
## Neon

Ma



```
config.properties
4 ### Application Config
5 scout.application.version=1.0.0
6 scout.application.name=Contacts (Migration Demo)
7
8 ### Service tunnel
9 scout.auth.publickey=MFYwEAYHKoZIzj0CAQYFK4EEAAoDQgAEFCjnU1MeOjGwsP7WQodjEPMJ8XdINDyS0HD8F
10
11 ### User Area
12 user.area=${user.home}/org.eclipsescout.contacts.server.dev
13
14 ### Database
15 contacts.database.schema.autocreate=true
16 contacts.database.data.autopopulate=true
17 contacts.database.jdbcMappingName=jdbc:derby:c:/DerbyDB/ContactDB;create=true;territory=en
18
19
20
21
22
23
24
25
26
27
28
29
30
31 ###
32 org
33 org
34
35
36 ###
37 org
38 org
39 org
40
41 org
42
43 ###
44 org
45 org.eclipsescout.contacts.server.services.common.sql.DerbySqlService#jdbcDriverName=org.ap
46 org.eclipsescout.contacts.server.services.common.sql.DerbySqlService#jdbcMappingName=jdbc:c
47 org.eclipsescout.contacts.server.services.common.sql.DerbySqlService#username=contact_user
48 org.eclipsescout.contacts.server.services.common.sql.DerbySqlService#password=secr3t
49
```

# Migrating «Contacts» DerbySqlService – pom.xml



```
org.eclipse.scout.contacts.parent/pom.xml
41 <build-helper-maven-plugin>
42 <property name="apache.derby.version" value="10.14.2.0" />
43 </properties>
44
45 <repositories>
61
62 <pluginRepositories>
78
79 <dependencyManagement>
80 <dependencies>
81 <dependency>
86 <dependency>
91 <dependency>
96 <dependency>
101 <dependency>
106 <dependency>
112 <dependency>
118 <dependency>
124 <dependency>
129 <dependency>
134 <dependency>
140 <!-- derby db needed for
141 <dependency>
142 <groupId>org.apache.derby</groupId>
143 <artifactId>derby</artifactId>
144 <version>${apache.derby.version}</version>
145 </dependency>
146

org.eclipse.scout.contacts.server/pom.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"
3 xmlns="http://maven.apache.org/POM/4.0.0"
4 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
5
6 <modelVersion>4.0.0</modelVersion>
7
8 <parent>
9 <groupId>org.eclipse.scout.contacts</groupId>
10 <artifactId>org.eclipse.scout.contacts.parent</artifactId>
11 <version>1.0.0-SNAPSHOT</version>
12 <relativePath>../org.eclipse.scout.contacts.parent</relativePath>
13 </parent>
14
15 <artifactId>org.eclipse.scout.contacts.server</artifactId>
16
17 <dependencies>
18 <dependency>
22 <dependency>
26 <dependency>
30 <dependency>
34 <dependency>
35 <groupId>org.apache.derby</groupId>
36 <artifactId>derby</artifactId>
37 </dependency>
38 </dependencies>
39 </project>
```

Overview Dependencies Dependency Hierarchy

Overview Dependencies Dependency Hierarchy Effective POM pom.xml



# Migrating «Contacts»

## DerbySqlService – ConfigProperties

### Mars

```
public class DerbySqlService extends AbstractDerbySqlService {}
```

### Neon

```
public class DerbySqlService extends AbstractDerbySqlService {  
  
    @Override  
    protected String getConfiguredJdbcMappingName() {  
        return CONFIG.getPropertyValue(  
            ConfigProperties.DatabaseJdbcMappingNameProperty.class);  
    }  
}
```

# Migrating «Contacts» Custom Services

## Mars

```
public class ContactService extends AbstractService  
    implements IContactService {  
    ...  
}
```

## Neon

```
public class ContactService implements IContactService {  
    ...  
}
```

# Migrating «Contacts» ServerApplication (startup logic)

## Mars

```
public class ServerApplication implements IApplication {  
  
    @Override  
    public Object start(IApplicationContext context) throws Exception {  
  
        @Override  
        public IStatus run(IProgressMonitor monitor) {  
            // Call all startup services to collect all available extensions  
            for (IServerStartupService service :  
                SERVICES.getServices(IServerStartupService.class)) {  
                service.init();  
            }  
        }  
  
        return Status.OK_STATUS;  
    }  
  
    return EXIT_OK;  
}
```

# Migrating «Contacts» ServerApplication (startup logic)

## Mars

```
public class ServerStartupService extends AbstractService
    implements IServerStartupService
{
    @Override
    public void init() throws ProcessingException {
        Set<String> tables = getExistingTables();
        createCompanyTable(tables, addInitialData());
        createContactTable(tables, addInitialData());

        LOG.info("DB install completed");
    }
}
```

# Migrating «Contacts» PlatformListener (startup logic)

## Neon

```
public class PlatformListener implements IPlatformListener {  
  
    @Override  
    public void stateChanged(PlatformEvent event) {  
        if (event.getState() == State.BeanManagerValid) {  
            autoCreateDatabase();  
        }  
    }  
  
    public void autoCreateDatabase() {  
        ...superUserRunContext.run(new IRunnable() {  
  
            @Override  
            public void run() throws Exception {  
                Set<String> tables = getExistingTables();  
                createOrganizationTable(tables);  
                createPersonTable(tables);  
            }  
        }));  
    }  
}
```

Migrating «Contacts»  
**Client**

# Migrating «Contacts»

## org.eclipsescout.contacts.client

### Just copy & paste

→ Lookup calls

### To modify for Neon

→ Move Icon image files

→ Forms & pages: «Pattern Matching»

– AbstractExtensibleFoo → AbstractFoo

– SERVICES.getService → BEANS.get

### Skipping

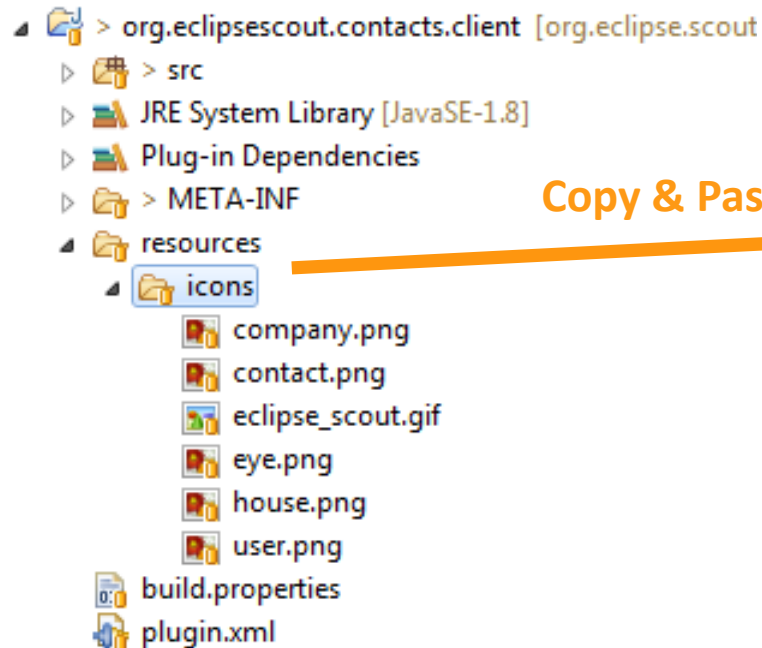
→ ClientSession

# Migrating «Contacts»

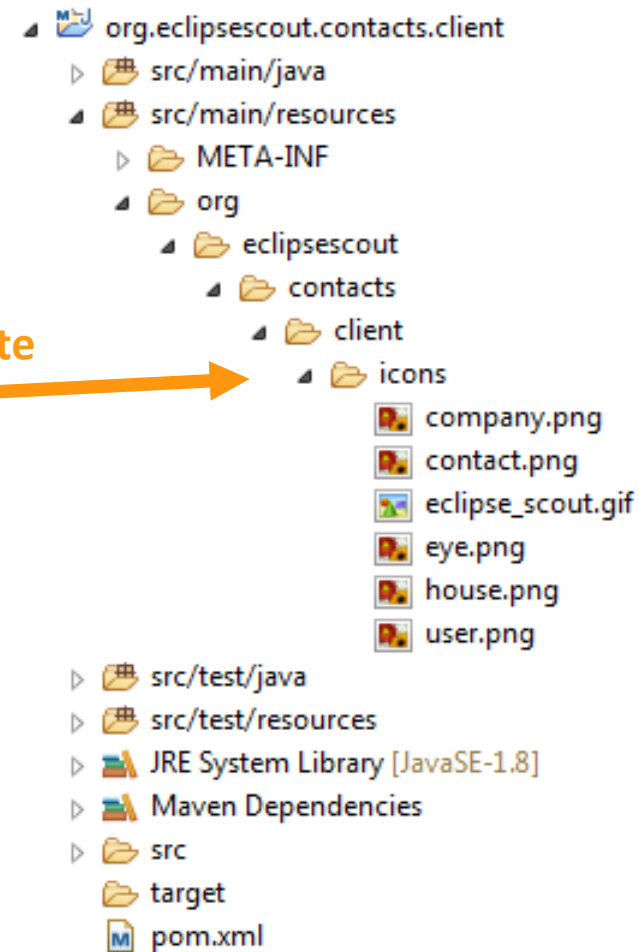
## Move Icon Images

### Neon

### Mars



Copy & Paste





# Migrating «Contacts» Client

## Neon

## Mars

```
@Order(1000.0)
public class EditURLMenu extends AbstractExtens:

    @Override
    protected String getConfiguredText() {
        return TEXTS.get("EditURL");
    }
}
```

```
public class ModifyHandler extends AbstractFormH:

    @Override
    protected void execLoad() throws ProcessingExc:
        IContactService service = SERVICES.getService
        ContactFormData formData = new ContactFormDa
        exportFormData(formData);
        formData = service.load(formData);
        importFormData(formData);
        setEnabledPermission(new UpdateContactPermis:

    }
}
```

```
@Order(1000.0)
public class EditURLMenu extends AbstractMenu {

    @Override
    protected String getConfiguredText() {
        return TEXTS.get("EditURL");
    }
}
```

```
public class ModifyHandler extends AbstractFormHandler {

    @Override
    protected void execLoad() throws ProcessingException {
        IContactService service = BEANS.get(IContactService.class);
        ContactFormData formData = new ContactFormData();
        exportFormData(formData);
        formData = service.load(formData);
        importFormData(formData);
        setEnabledPermission(new UpdateContactPermission());

    }
}
```

# Migrating «Contacts» Demo

# «Contacts» Migration Demo

The screenshot shows a web browser window with the address bar displaying 'localhost:8082'. The browser's menu bar includes 'File', 'Favorites', and 'Help'. The main content area displays a contact form titled 'Contact'. On the left side of the form is a black and white illustration of a woman in a long dress and hat. The form fields are as follows:

First Name	<input type="text" value="Alice"/>
Last Name	<input type="text"/>
Date of Birth	<input type="text" value="11/26/1865"/>
Gender	<input type="radio"/> Male <input checked="" type="radio"/> Female
Street	<input type="text"/>
Phone	<input type="text"/>
Location	<input type="text" value="Daresbury, Che:"/> <input type="text" value="GB:"/> <input type="button" value="Q"/>
Mobile	<input type="text"/>
Email	<input type="text"/>

Below the form, there are three tabs: 'Details', 'Work', and 'Comments'. The 'Details' tab is active. At the bottom of the form, there are 'OK' and 'Cancel' buttons. The 'OK' button is highlighted in dark blue. At the bottom of the browser window, there is a search bar, a summary bar showing '2 rows loaded' and 'One row selected', and buttons for 'Reload data' and 'Select all'.

# Migration to Neon Tips



## Test and Check Early

- Play with new sources early
- Are all your essential features implemented?



## Plan Ahead

- Team may not be able to work as usual during migration
- Merges may be more difficult (project structure)



## Plan Knowhow Transfer

- Be aware of new concepts in Scout Neon!
  - Check out the docs (book, forum) [1]
  - BSI Offers 2 day Upgrade Course (Mars -> Neon)

[1][https://tools.bsiag.com/scoutbook/5.2/latest/book\\_scout\\_architecture/scout\\_architecture/scout\\_architecture.html](https://tools.bsiag.com/scoutbook/5.2/latest/book_scout_architecture/scout_architecture/scout_architecture.html)



## Need Help?

- BSI offers migration support



**Summary**

# Migrating to Neon

## Summary

### «Contacts»

- Technical migration, straight forward
- No change for the majority of components
- Includes refactoring, usage of new features

### General:

- Positive feedback from migration of BSI CRM to Neon
- Migration tooling depending on additional experience
- We are here to help 😊

# Thanks

@EclipseScout 

{judith.gull | matthias.zimmermann}@bsi-software.com