Eclipse 4

Eclipse Day Toulouse 24 mai 2012

OPC 12 ECD PRE E4A 01 A

Table of contents

Eclipse 4

Introduction / OPCoach

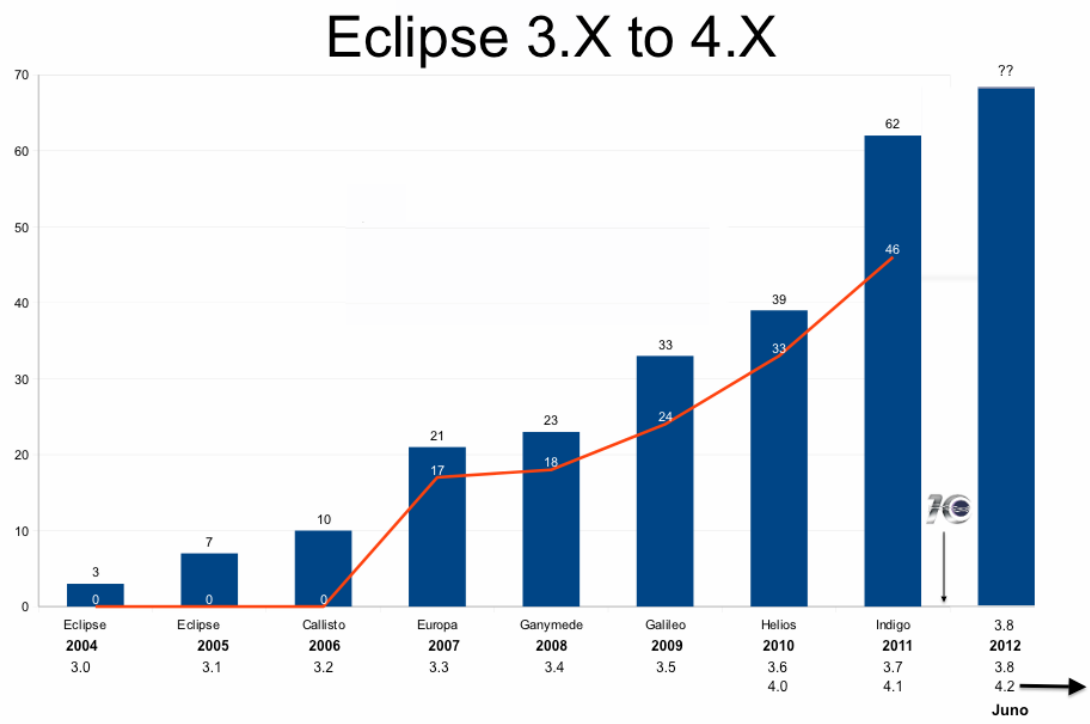
- Company founded in June 2009 *http://www.opcoach.com*
- Member of the Eclipse Foundation (Solution Member)
- 3 Eclipse-based activities:
  - Expertise
  - Training
  - Employment

Agenda

This talk will focus on :

- 3.X history, advantages and disadvantages
- E4 architecture: the application model
- Injection and annotations
- CSS Styling
- Compatibility layer
- Questions : keep 3.X ? use 4.X ?

3.X deliveries



Eclipse 3.X History

Reasons for change

The benefits of Eclipse 3.X / RCP:

- rich software platform
- architecture based on OSGi
- extension point concept
- portable to all environments
- advanced Tooling (JDT, PDE, EMF, ...)
- adopted by many providers
- EPL : open source license

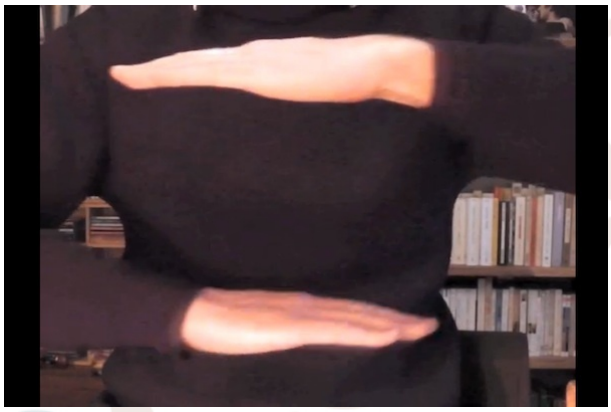
But ...

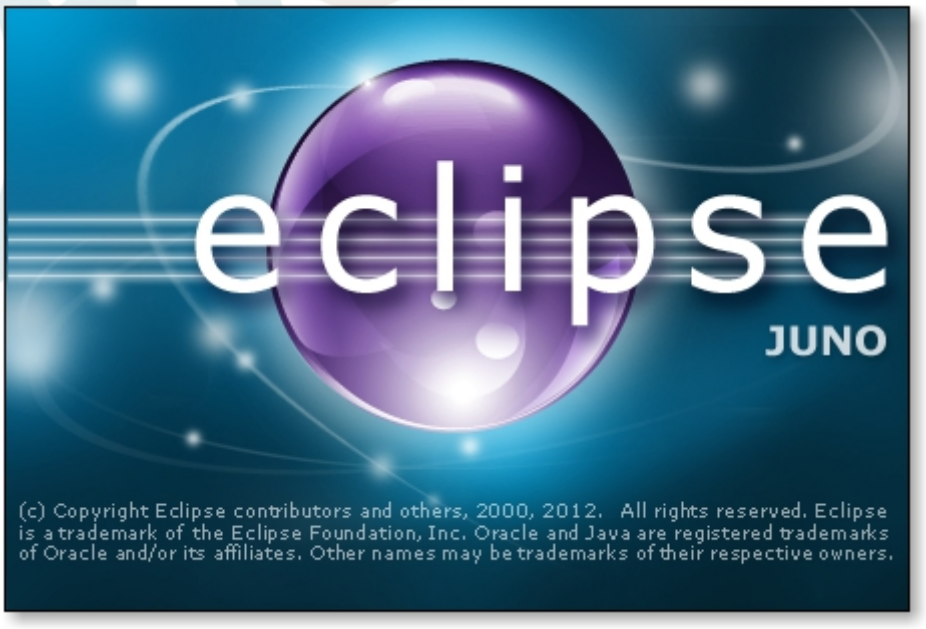
Eclipse 3.X / RCP drawbacks

- 'old' software (8 years)
- many extension points and extensions dispersed
- the developed application depends largely on the framework (inheritance, dependencies ...)
- not always consistent API
- many singletons
- injection mechanisms are not implemented
- no separation of content and appearance (no renderer)
- no CSS : difficult to parametrize the UI
- low usage of OSGi services

Change

... is now!





Juno

E4 vs. E3

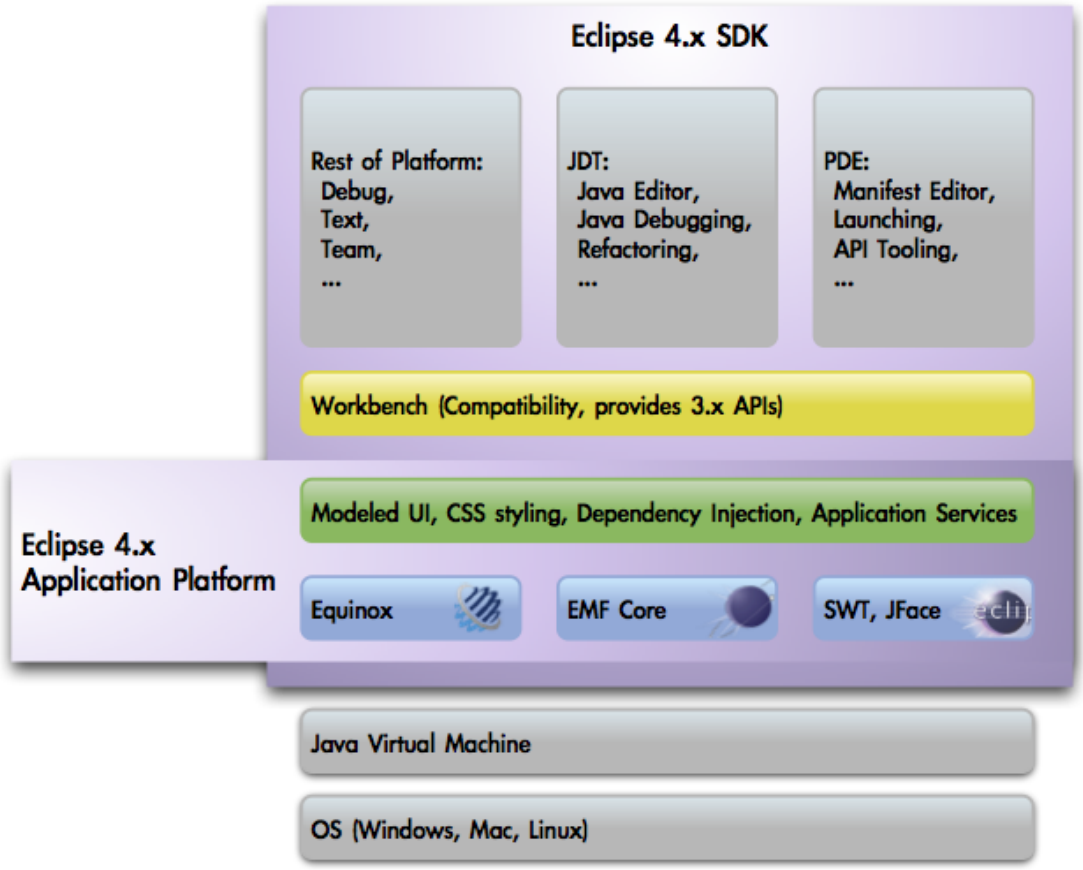
strengths of E4

- Definition of an application model : centralizes all components of the application
- Unified concepts of UI components : view, editor, perspective is no more mandatory
- Support injection and annotations
- CSS Styling
- Compatibility layer

What is kept from E3:

- some extension points (those that are not managed by the application model)
- SWT / JFace and the business code
- architecture principles: bundle, plugins, fragments

e4 Architecture



e4 Architecture

What will change in the code:

- accessing the platform (using the injection)
- removing many inheritance relations (ViewPart, AbstractHandler, etc ...)
- moving some extensions to the model (views, handlers..)

- services : ESelectionService, EMenuService,
- the CSS management

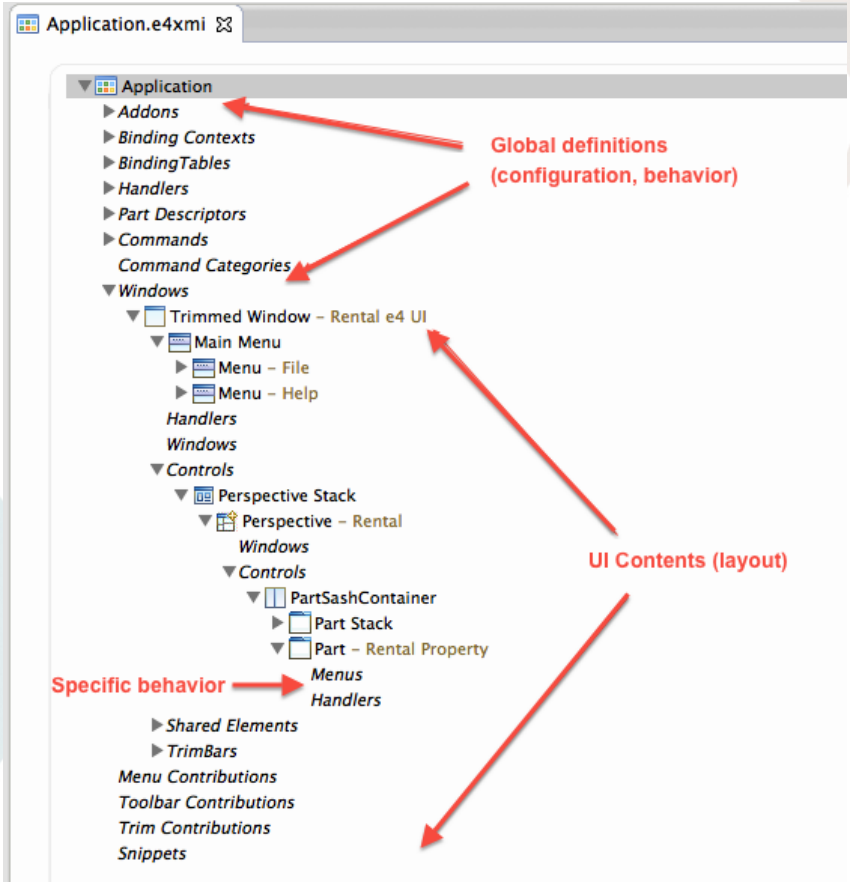
A. Application Model

The application model

This is a global model that gathers the usual extension points :

- view, perspective
- commande, handlers, menu

The application model



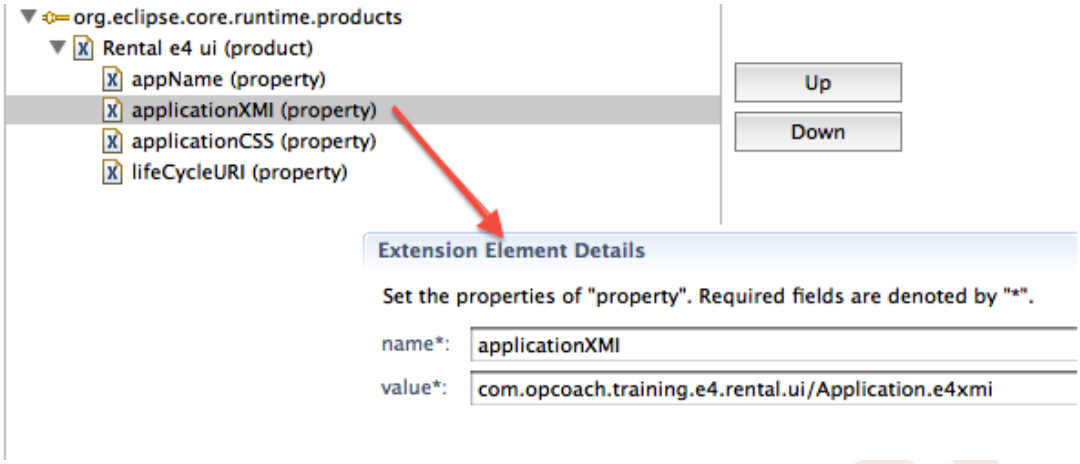
Application Model

The contents of pure UI or commands are defined by the code

Use Cases

In the case of a 3.X application running on 4.X engine, the model is filled by the compatibility layer.

In the case of an 4.X application the model is associated to a product (using extension point):



product extension point

Use cases

In all cases:

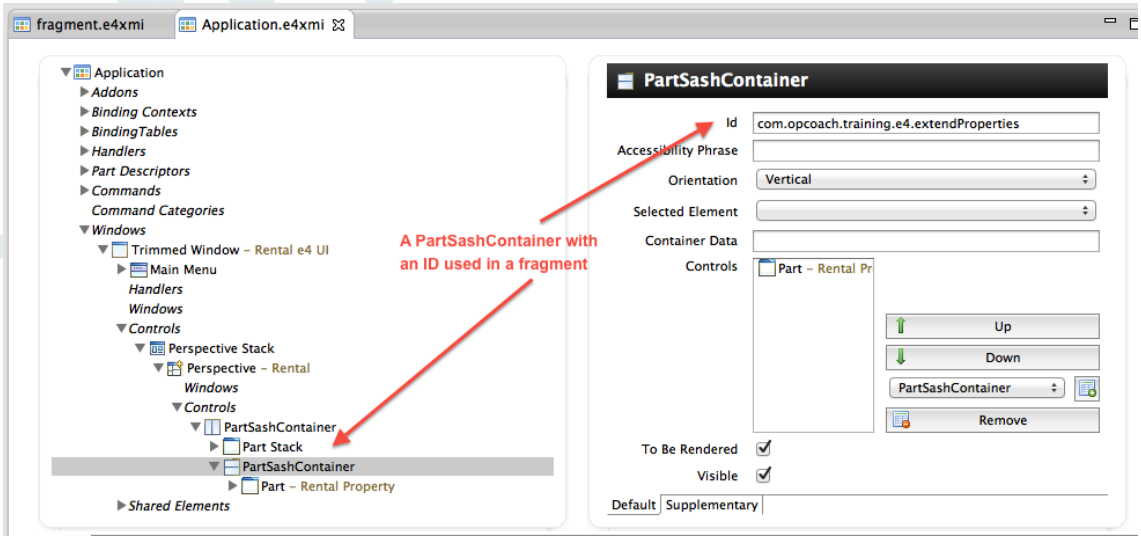
- The model is interpreted dynamically -> any changes will be released on the UI
- You can view the current model with the Alt Shift F9 shortcut

Model fragment

The fragment model behaves similarly to a fragment plugin
It complements an existing application model
The links are made by matching ID

Model fragment

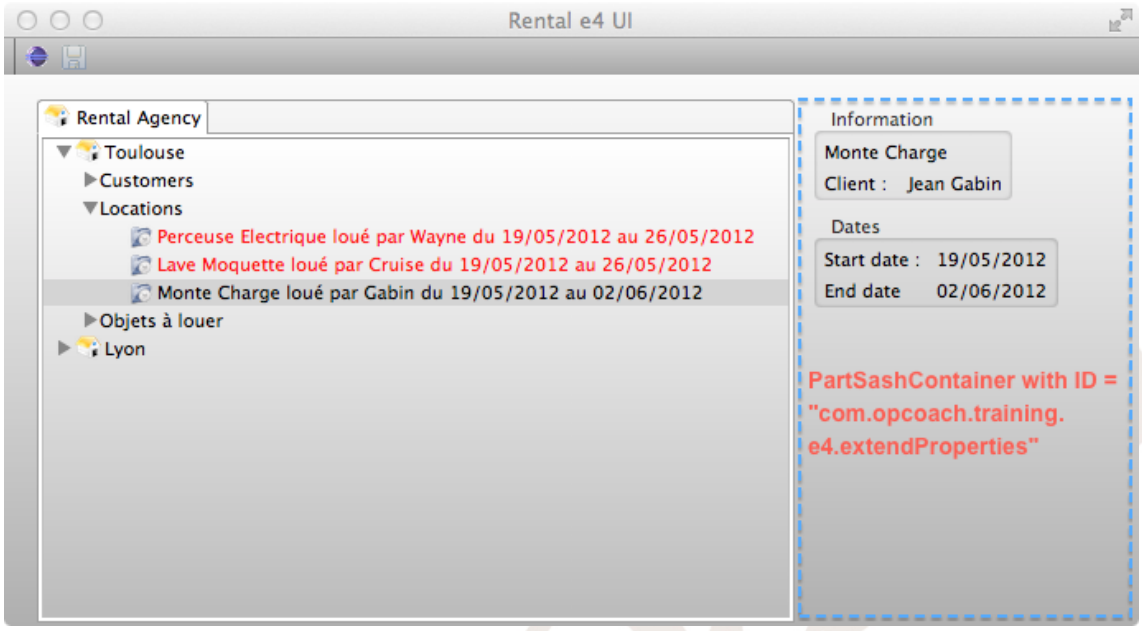
Locate the ID of a model component that will be completed :



Model sample

Model fragment

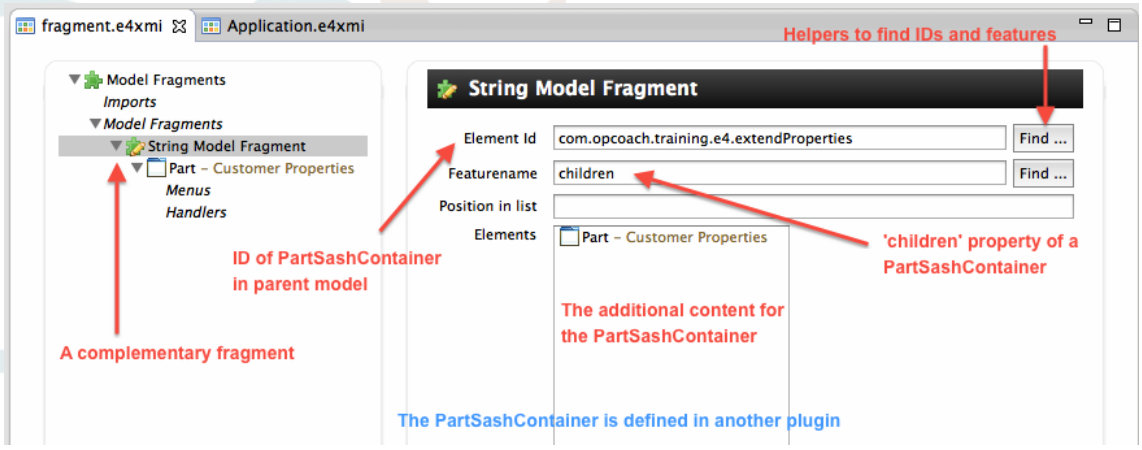
The model without the fragment, produces this application:



UI without fragment

Model fragment

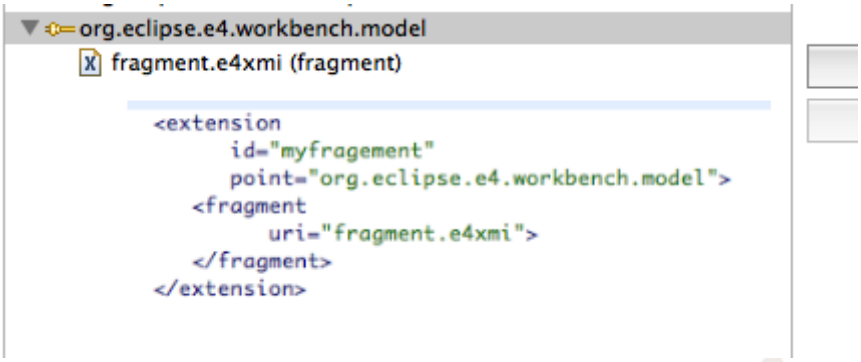
Referencing this ID in the fragment (created in another plugin):



Model Fragment

Template fragment

Adding the fragment in an extension (**org.eclipse.e4.workbench.model**)



Model Fragment

Model fragment result

UI with fragment

B. E4 injection and annotations

Introduction / Principle

The injection mechanism delegates to a framework the initialization of class fields or method parameters.

Using the @ Inject annotation applied to a constructor, method or field.

Use:

The GUI renderer of E4, will call the Parts by instantiating and injecting the expected values.

```
15
16 public class SampleView
17 {
18     // This field is initialized by injector
19     @Inject
20     private ESelectionService selectionService;
21
22     // Constructor called with 2 injected parameters, second is optional
23     @Inject
24     public SampleView(Composite parent, @Optional IStylingEngine styleEngine)
25     {
26         // Code to create the view...
27     }
28
29 }
30
```

Inject sample

The framework introspects classes and finds the methods, constructors and annotations (@ Inject ...)

Injection: advantages / disadvantages

Benefits :

- significant reduction in the coupling (the inheritance with the framework is no longer necessary)
- the injector gathers all shared objects (hierarchical contexts)
- the dependencies are only on very high level interfaces

Disadvantages :

- when debugging an injected method, the caller code is introspection code
- requires knowledge of the objects that can be injected
- must master the lifecycle of the framework:
  - see: *http://www.vogella.com/articles/Eclipse4RCP/article.html* : Chapter 16 Behavior
- there is less code control when editing:
  - classes are referenced by name in non-java editors (application model editor)
  - errors can be found only at runtime (no message 'never initialized object')

What can you inject?

- it depends on where you are.
- Eclipse e4 creates the contexts when needed (creating a view, gains focus ...).
- See: *http://wiki.eclipse.org/Eclipse4/RCP/EAS/List_of_All_Provided_Services*

Other annotations

Annotations JSR 330 standard

- **@ Named** : For injecting an object by its name
- **@ Singleton** : To indicate that a class should be instantiated only once
- **@ Provider <T>** : Delegates the construction of objects to a Provider

E4AP specific annotations

- **@ Optional** : Injects a null if no object is found (no exception thrown)
- **@Active** : Retrieves the current active part
- **@ Preference** : Used to inject the value stored in the preferences
- **@ Creatable** : Allows injection of a non-existent object in the injector

Example:

```
116    @Inject
117    public void setSelection(@Optional @Named(IServiceConstants.ACTIVE_SELECTION) Object o,
118                             Adapter adapter)
119    {
120        Rental r = adapter.adapt(o, Rental.class);
121        setRental(r);
122
123    }
124
```

@Inject @Named @Optional

Other annotations used in e4

Annotations set on methods:

- **@ PreDestroy** : Called before deleting an instance
- **@ PostConstruct** : Called after instanciation and all fields injected
- **@ Focus** : Called when the Part gets focus
- **@ Persist** : Called to handle the storage of an editor
- **@ Execute** : Called to execute a Handler
- **@ CanExecute** : Called to check if a handler can run

And more annotations (for the life cycle of the application):
@ PostContextCreate, ProcessAdditions @, @ ProcessRemovals, Presave @, @ EventTopic ...

Final example code of 'Part'

```
16 public class SampleView
17 {
18     @Inject
19     private ESelectionService selectionService;
20
21     @Inject
22     private EMenuService menuService; // This field is initialized by injector
23
24     @Inject
25     public SampleView(Composite parent, @Optional IStylingEngine styleEngine)
26     {
27         // Code to create the view...
28     }
29
30     @Inject
31     public void anotherMethod(@Optional EMenuService service)
32     {
33          // This method will be called
34     }
35
36     @PostConstruct
37     void initializeListeners(@Active MPart part)
38     {
39         // A method called after all injected method have been called
40     }
41
42     @PreDestroy
43     public void dispose()
44     {
45         // Call this method before deleting object;
46     }
47
48     @Focus
49     public void onFocus()
50     {
51         // This method is called when part takes focus
52     }
53
54 }
55
```

A viewPart with annotations

Annotations: advantages / disadvantages

Benefits :

- The code is simplified
- An annotation simplifies the code: if we use the selection, you will be notified when it changes
- Associated with injection, we can centralize the needs of a method in its API:
  - ex: if the focus needs the service selection: @Focus void myFocus (ESelectionService serv)

Disadvantages :

- annotations are not always intuitive
- many annotations defined by different frameworks (e4, ejb, java, ...)
- one must know the annotations to use and where to apply
  - search the descendants of **javax.lang.Annotation**
- annotated classes derive no more from high-level classes (harder to understand)
- the java editor does not know what is a class at the start: it is understood by seeing the annotations
  - if @ Persist inside, it is an editor, @ Execute is used to define an Handler
- we can forget to implement necessary methods for a class
  - if we do not put the @ Persist editor does not save its contents, or @ Execute for the handler it does not run

C. CSS Styling

In the java code

E4AP has its own CSS rendering engine

Refering to a css class in the source code :

```
25    @Inject
26    public SampleView(Composite parent, @Optional IStylingEngine styleEngine)
27    {
28        // Code to create the view...
29        TreeViewer agencyViewer = new TreeViewer(parent);
30        // ...more code
31
32
33        // Add the e4 styling
34        if (styleEngine != null)
35        {
36            styleEngine.setClassname(agencyViewer.getControl(), "agencyViewer");
37        }
38    }
```

Styling in Java

Content of the css

Writing a CSS:

default.css

```
1
2
3  .MTrimmedWindow.topLevel {
4      margin-top: 15px;
5      margin-bottom: 2px;
6      margin-left: 20px;
7      margin-right: 20px;
8      background-color: #FFF #AAA 100%
9  }
10
11 .MTrimBar {
12     background-color:  #CFCFCF #A8A8A8 100%;
13 }
14
15 .agencyViewer {
16      background-color: #FFFFFF #0000FF 100%
17 }
```

Use UI Model java class

Use the CSS class defined in the java code

Css

Using the CSS

The CSS is defined in the product extension point :

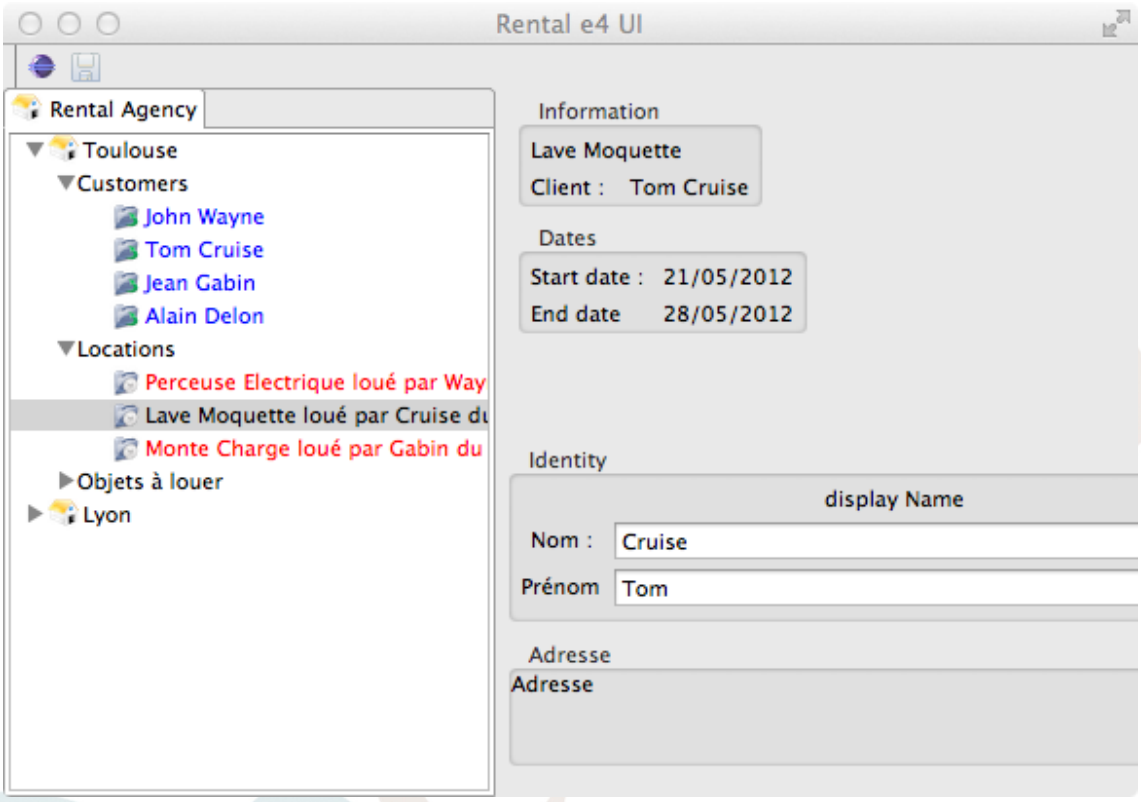
com.opcoach.training.e4.rental.ui

```
37  <extension
38        id="product"
39        point="org.eclipse.core.runtime.products">
40     <product
41           application="org.eclipse.e4.ui.workbench.swt.E4Application"
42           description="Rental ui made over e4"
43           name="Rental e4 ui">
44        <property
45              name="appName"
46              value="Rental e4">
47        </property>
48        <property
49              name="applicationXMI"
50              value="com.opcoach.training.e4.rental.ui/Application.e4xmi">
51        </property>
52        <property
53              name="applicationCSS"
54              value="platform:/plugin/com.opcoach.training.e4.rental.ui/css/default.css">
55        </property>
56     </product>
57  </extension>
58
```

product extension

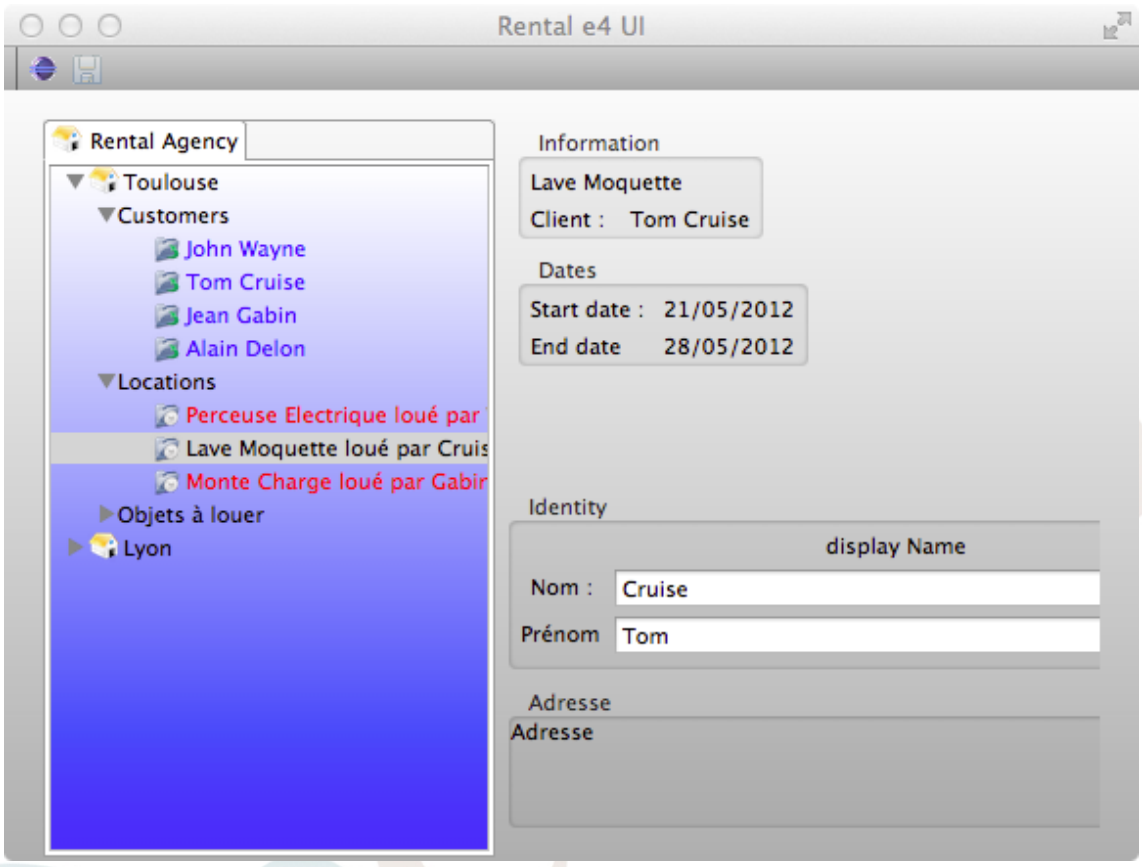
Result of CSS

Without CSS:



without css

CSS:



with css

D. Compatibility Layer

Introduction

E4 comes with a compatibility layer that allows:

- to launch an E3 application on the E4 platform
- to gradually migrate the application

Usage

Install Eclipse 4.2

Install an Eclipse 3.X projects in the workspace

Create a specific launch configuration :

- select your application's main plugin
- add the missing plugins:
  - **org.eclipse.equinox.ds**
  - **org.eclipse.equinox.event**
  - **org.eclipse.platform**
- To view the application model line (Alt Shift F9), also add:
  - **org.eclipse.e4.tools.emf.liveeditor**
- add required plugins

From 3.X to 4.X

- Create an application model : application.e4xmi
  - following the application model obtained with Alt Shift F9
  - or using the **LegacyIDE.e4xmi** (In org.eclipse.ui.workbench)
- Modify the code:
  - views and editors: removing inheritance on ViewPart, using injection
  - handlers: declaring commands and handlers in the model
  - advisors: moving the construction of menus in the model
  - selection: using ESelectionService and injection
  - changing the access to Platform, PlatformUI, ...
  - extracting the specific graphic in CSS
- Keeping :
  - all the business code (EMF, non UI specific code) and test unit
  - all non UI extensions (adapters, expression definition ...)
  - architecture with plugins and fragments
  - internationalization

E. Summary

What to do?

Your application is developed on 3.X

- it is almost finished -> keep the 3.X and go for 3.8
- it is under development, according to the progress, you can :
  - keep the 3.X and schedule later a migration
  - continue new developments using 4.X and 3.X code
  - bring the existing 4.X and migrate to this new version definitely

The development of your application has not yet started:

- You are pioneers : use eclipse 4.X !
- You have time constraints and want to be sure of 4.X: use eclipse 3.8 and schedule a gradual migration on 4.3

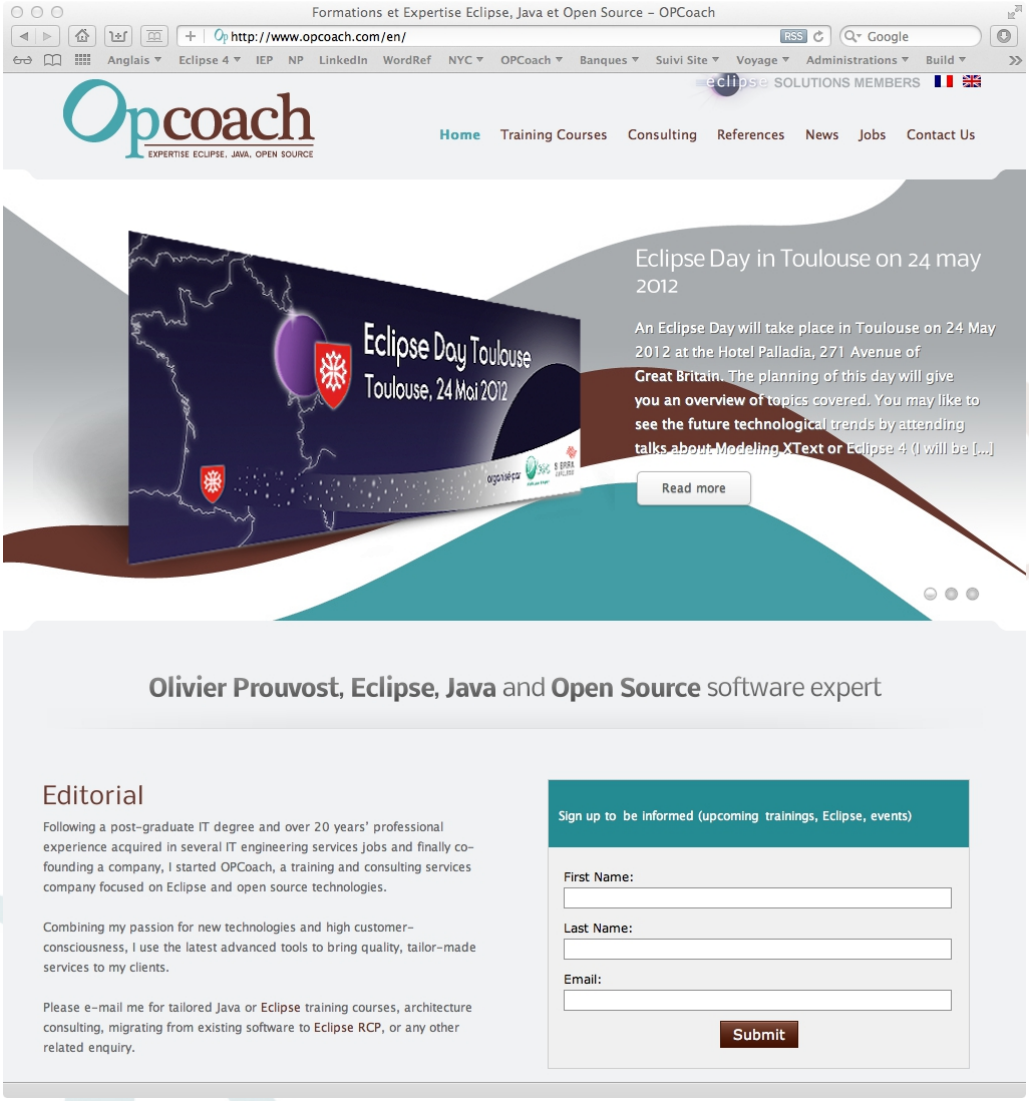
IN ALL CASES : Use Eclipse 4.2 as IDE in June 2012!

References

- Eclipse 4 home site : *http://eclipse.org/eclipse4/*
- Eclipse 4 forum : *http://www.eclipse.org/forums/index.php?t=thread&frm_id=12*
- Wiki Eclipse 4 : *http://wiki.eclipse.org/Eclipse4/RCP*
- Tutorial Lars : *http://www.vogella.com/articles/Eclipse4RCP/article.html*
- Tutorial Tom : *https://github.com/tomsontom/e4demo/raw/master/tutorial.pdf*
- Eclipse 4 DI : *http://wiki.eclipse.org/Eclipse4/RCP/Dependency_Injection*

To receive the pdf of this talk :

Register on opcoach mailing list on the home page of *http://www.opcoach.com*


Formations et Expertise Eclipse, Java et Open Source – OPCoach
http://www.opcoach.com/en/
Google
Anglais
Eclipse 4
IEP
NP
LinkedIn
WordRef
NYC
OPCoach
Banques
Suivi Site
Voyage
Administrations
Build
SOLUTIONS MEMBERS
Opcoach
EXPERTISE ECLIPSE, JAVA, OPEN SOURCE
Home
Training Courses
Consulting
References
News
Jobs
Contact Us
Eclipse Day Toulouse
Toulouse, 24 Mai 2012
Eclipse Day in Toulouse on 24 may 2012
An Eclipse Day will take place in Toulouse on 24 May 2012 at the Hotel Palladia, 271 Avenue of Great Britain. The planning of this day will give you an overview of topics covered. You may like to see the future technological trends by attending talks about Modeling XText or Eclipse 4 (I will be [...]
Read more
Olivier Prouvost, Eclipse, Java and Open Source software expert
Editorial
Following a post-graduate IT degree and over 20 years' professional experience acquired in several IT engineering services jobs and finally co-founding a company, I started OPCoach, a training and consulting services company focused on Eclipse and open source technologies.
Combining my passion for new technologies and high customer-consciousness, I use the latest advanced tools to bring quality, tailor-made services to my clients.
Please e-mail me for tailored Java or Eclipse training courses, architecture consulting, migrating from existing software to Eclipse RCP, or any other related enquiry.
Sign up to be informed (upcoming trainings, Eclipse, events)
First Name:
Last Name:
Email:
Submit