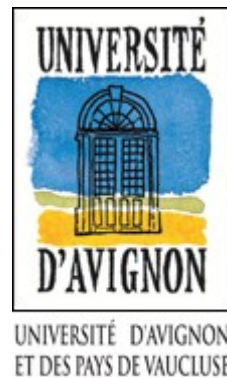


Prototypage et Interfaces utilisateurs



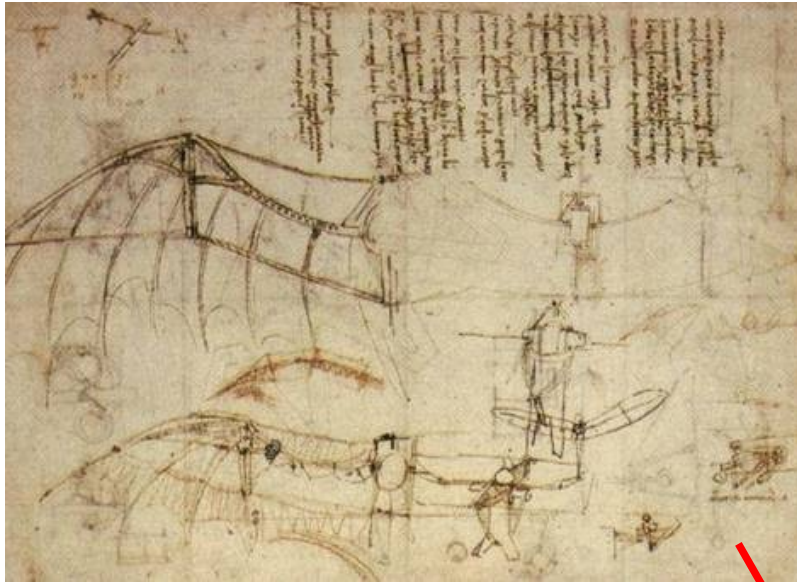
GL Avancé: Prototypage et interfaces utilisateur

2 / 51

Un peu d'histoire – Les prototypes

- 1488 : La machine volante de Léonard De Vinci : Processus d'invention

Plans



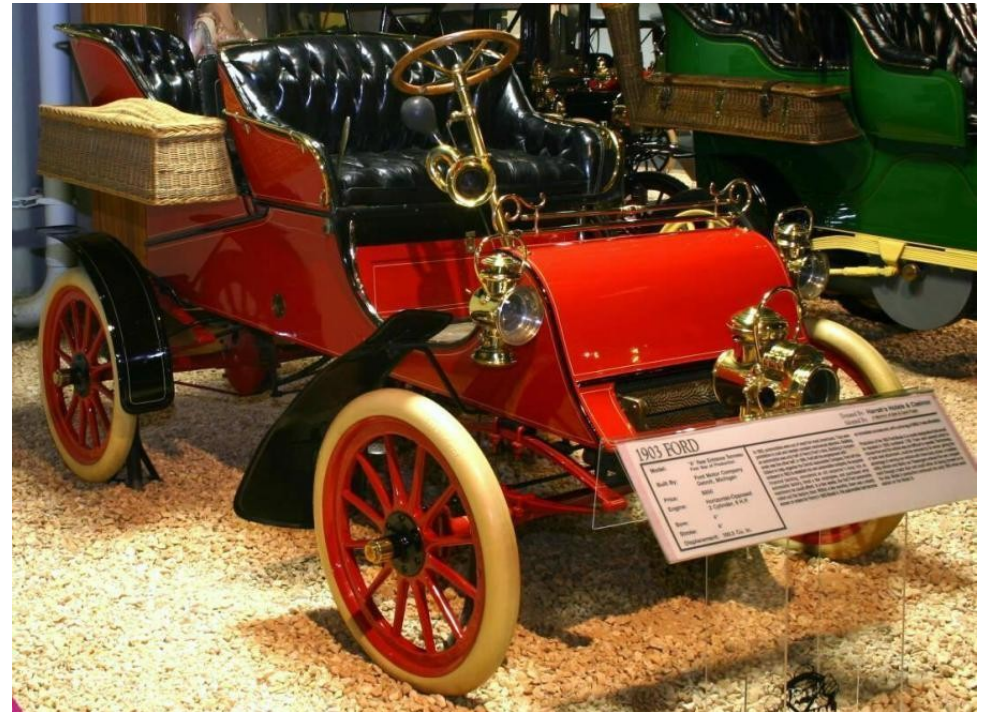
Prototype

GL Avancé: Prototypage et interfaces utilisateur

3 / 51

Un peu d'histoire – Les prototypes

- 1886: Karl Benz : La première voiture 3 roues, motorisée à essence

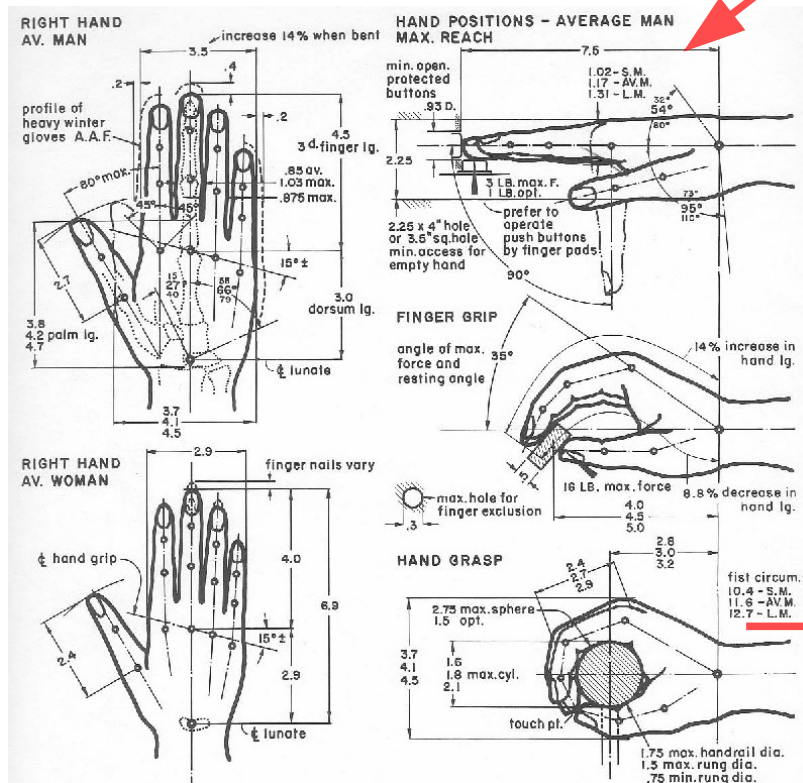


- 1903 : Ford : Voiture avec 4 roues et une direction à crémaillère

GL Avancé: Prototypage et interfaces utilisateur

Un peu d'histoire – Les prototypes

- 1937 : Le téléphone d'Henry Dreyfuss : Prototype ergonomique centré sur l'utilisateur



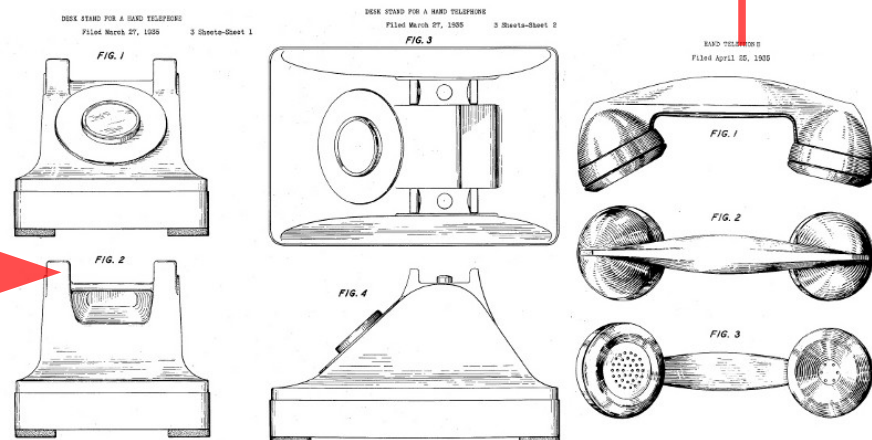
Mesures des mains



Hand Phone (1877 : Bell)



Prototype



Design

GL Avancé: Prototypage et interfaces utilisateur

5 / 51

Un peu d'histoire – Les interfaces

- 1728 : Carte perforée pour métiers à tisser



- 1928 : Carte perforée pour ordinateur IBM



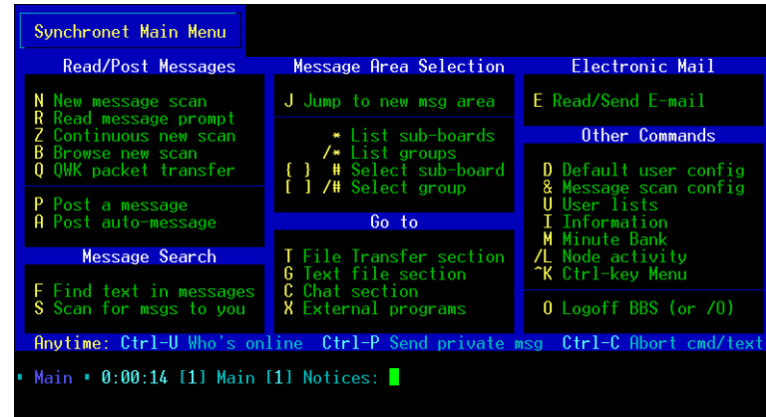
- 1960 : Ligne de commande



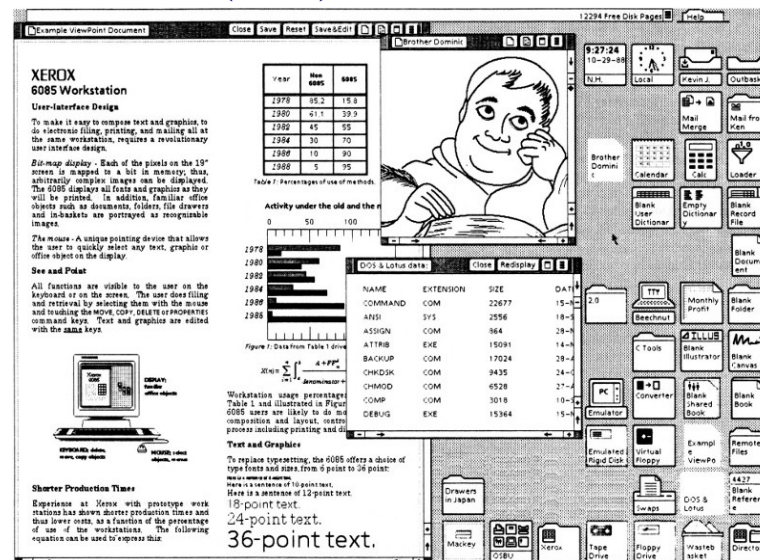
GL Avancé: Prototypage et interfaces utilisateur

Un peu d'histoire – Les interfaces

➤ 1970 : Textual User Interface (TUI)



➤ 1980 : Graphical User Interface (GUI)



GL Avancé: Prototypage et interfaces utilisateur

7 / 51

Définitions

- **Prototype:**

- Le prototype est un exemplaire incomplet et non-définitif de ce que pourra être le produit ou l'objet final.

- **Prototypage:**

- La démarche (ou méthode) consistant à réaliser un prototype. Le prototypage est d'autant plus efficace qu'il est itératif.

- **Prototypage d'interface utilisateur:**

- Le prototypage (ou maquettage) d'interface utilisateur est la méthode de conception des interfaces en ergonomie informatique. Le prototype matérialise l'interface homme-machine du logiciel, et sert de référence aux différents acteurs intervenants dans la phase de spécification d'un projet informatique.

GL Avancé: Prototypage et interfaces utilisateur

8 / 51

Objectifs du cours

- **Prototypage simple (dit horizontal):**
 - Conception d'une maquette statique (mockup)
 - Traduction des besoins utilisateur en composants graphiques
- **Prototypage fonctionnel (dit vertical):**
 - Ajout d'interactivité : maquette dynamique
 - Binding du modèle de données avec l'interface
 - Réalisation d'un scénario d'utilisation (use case)
- **Prototypage d'échelle sur une plateforme dédiée:**
 - Utilisation d'un environnement de fenêtrage de haut niveau
 - Découpage en plugins
 - Branding et personnalisation
 - Déploiement et livraison du prototype

GL Avancé: Prototypage et interfaces utilisateur

9 / 51

Supports

- **Prototypage d'une application de gestion de contacts**

- Logiciel de type « client lourd »
- Paradigme MVC : Modèle-Vue-Contrôleur

- **Langage de programmation**

- Java SDK (version 8u40 ou +)
- API Swing (JFrame, JDialog, JList, JTable ...)
- API JavaFX (Stage, Scene, Dialog, ListView, TableView, ...)



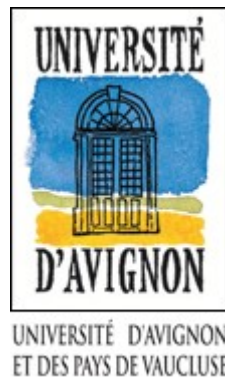
- **Outils de prototypage et de développement**

- NetBeans IDE (version 8.1)
- Scene Builder (version 8.1)



1ère partie:

Prototypage simple



GL Avancé: Prototypage et interfaces utilisateur

11 / 51

Prototypage simple

- **Prototype horizontal:**

- Maquette statique
- Confirmation des exigences de l'interface utilisateur
- Réduction des risques d'erreurs ergonomiques
- Version de démonstration pour obtenir des retours du client
- Première estimation du développement : temps, coût, charge de travail
- Support efficace à la vente et à la communication autour du futur produit

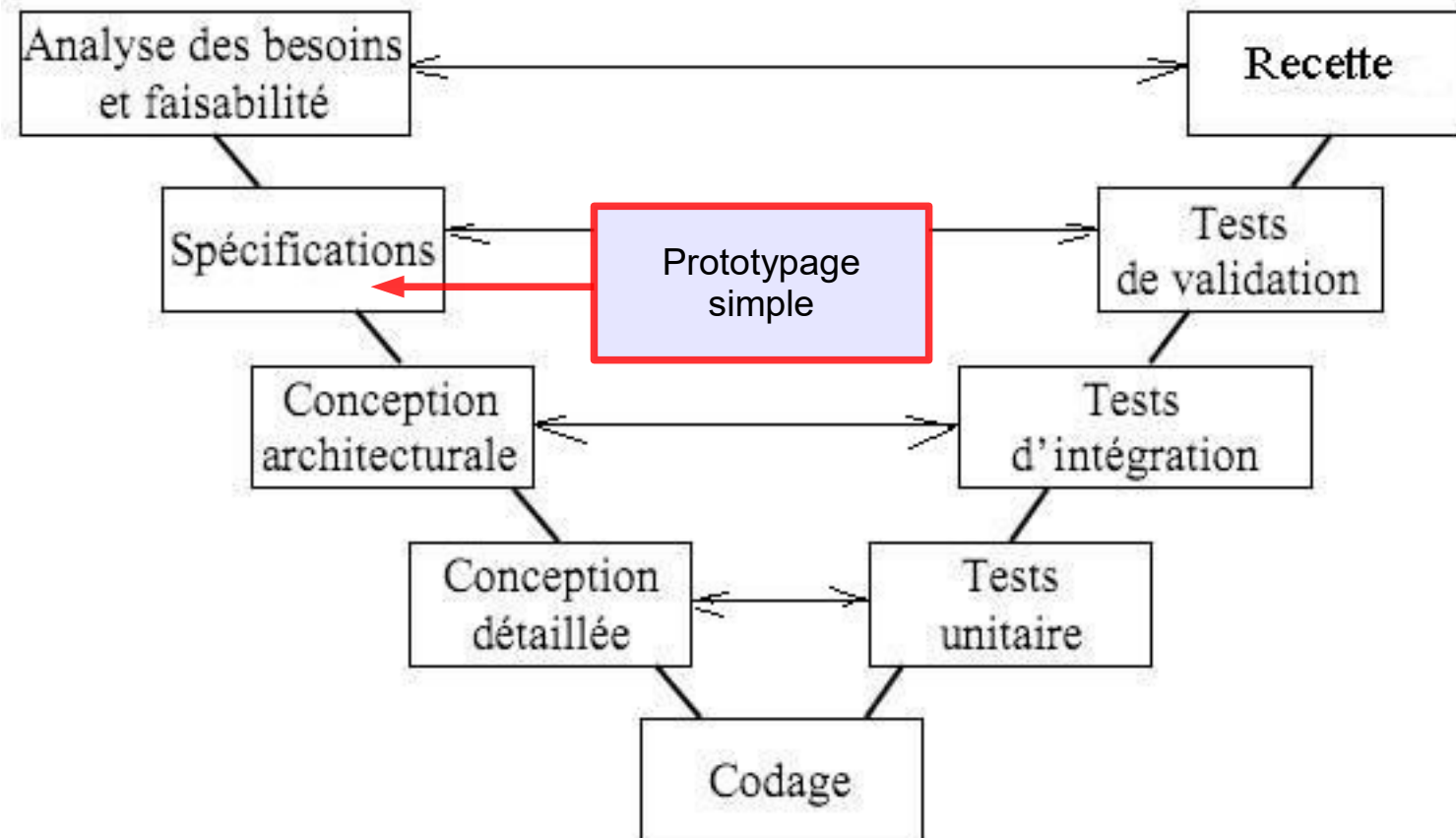
→ **Étape importante de la conception d'une application ergonomique**

GL Avancé: Prototypage et interfaces utilisateur

12 / 51

Cycle de vie d'une application

- Modèle en V: démarche incrémentale**



GL Avancé: Prototypage et interfaces utilisateur

13 / 51

Conception itérative d'une maquette

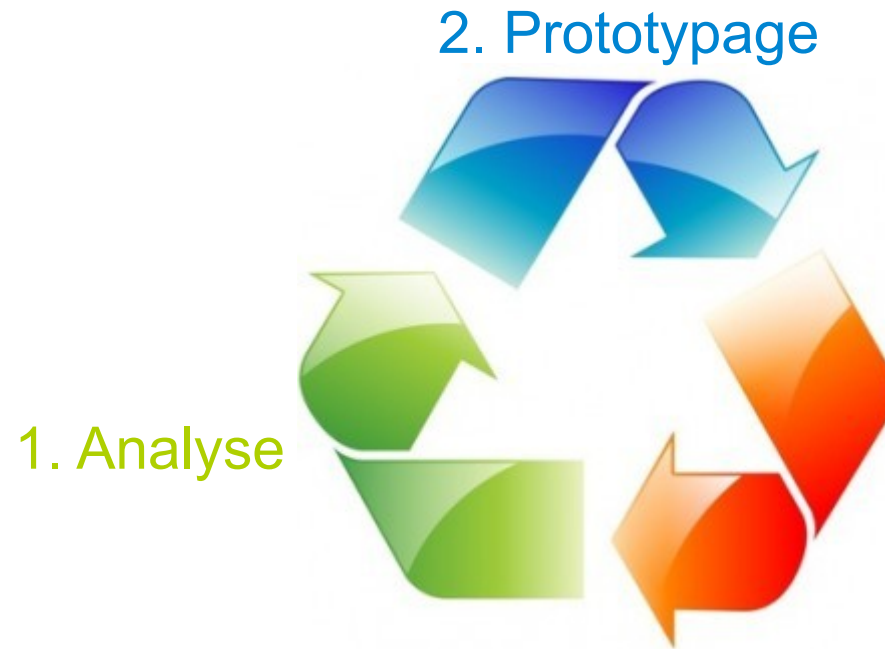


- **Analyse:**
 - ✓ Analyser les besoins et les principales fonctions
 - ✓ Identifier le type d'utilisateur (client, administrateur, décideur, expert, ...)
 - ✓ Spécifier les cas d'utilisation

GL Avancé: Prototypage et interfaces utilisateur

14 / 51

Conception itérative d'une maquette



- **Prototypage:**

- ✓ Réaliser une maquette informelle : schématiser les écrans
- ✓ Simuler les fonctionnalités attendues (ne pas tout implémenter!)
- ✓ Ajuster l'ergonomie de l'interface selon l'utilisateur
- ✓ Préciser la cinématique avec le client

GL Avancé: Prototypage et interfaces utilisateur

15 / 51

Conception itérative d'une maquette



- **Évaluation:**

- ✓ Mesurer le niveau de satisfaction de l'utilisateur
- ✓ Évaluer la pertinence de l'ergonomie
- ✓ Collecter les remarques des testeurs
- ✓ Tester la réalisation et l'intégration de tous les cas d'utilisation

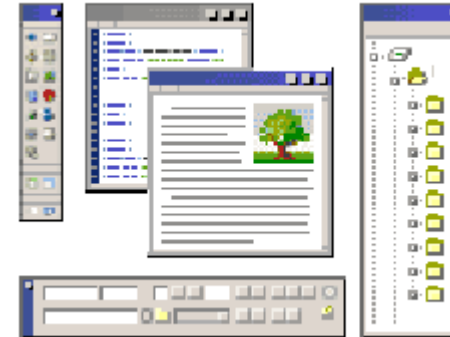
GL Avancé: Prototypage et interfaces utilisateur

16 / 51

Prototypage : mode de fenêtrage

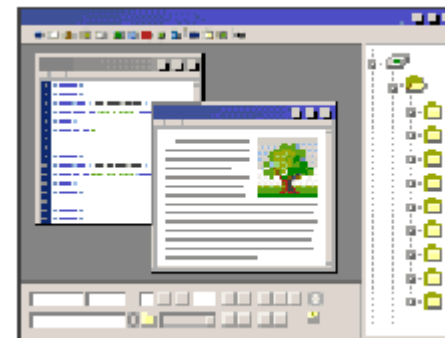
- **MDI: Multiple Document Interface**

- Gestion de plusieurs fenêtres « volantes »
- Pas de conteneur principal
- Interface très modulable
- Espace de travail optimisé et superposable
- En Swing : JDesktopPane et JInternalFrame



- **SDI: Simple Document Interface**

- Gestion d'une unique fenêtre
- Conteneur global de l'application
- Interface figée
- Effort de conception ergonomique
- En Swing : JFrame et JDialog



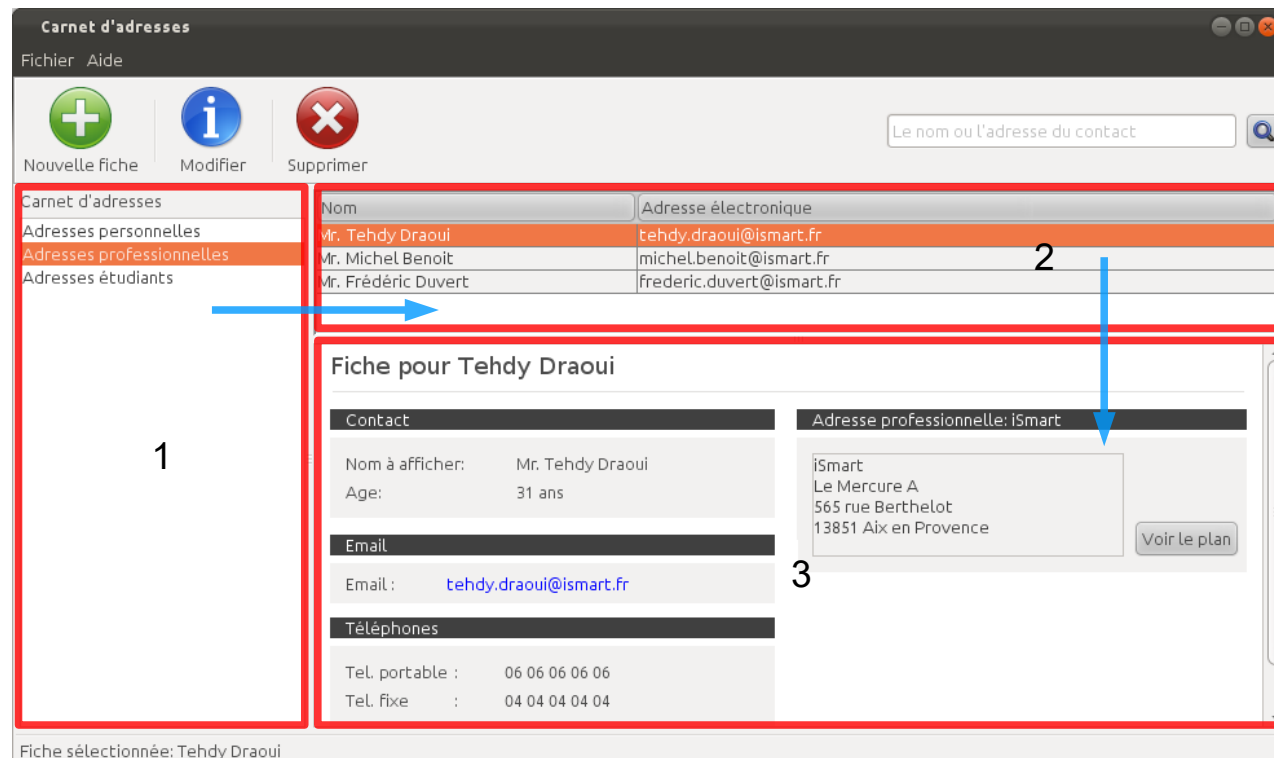
GL Avancé: Prototypage et interfaces utilisateur

17 / 51

Réalisation d'une maquette

- **Définition des conteneurs de haut niveau:**

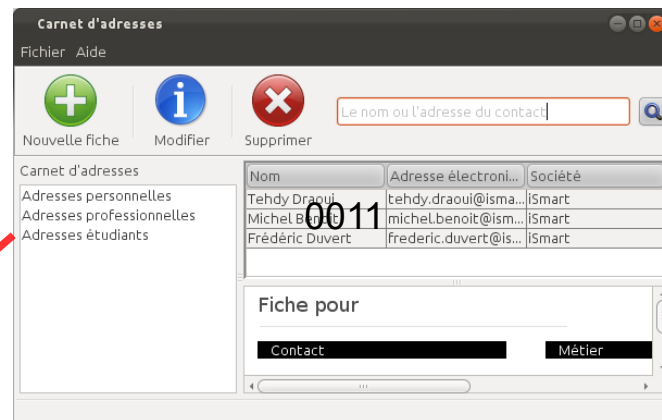
- ✓ Découpage en vues logiques
- ✓ Cinématique de l'application
- ✓ Disposition ergonomique (zoning)
- ✓ Système de navigation intuitif



GL Avancé: Prototypage et interfaces utilisateur

Disposition des composants

- Disposition statique (Null layout) : composant positionné par X, Y, Width et Height
- Disposition dynamique : layout avec règles de distribution de l'espace libre



statique

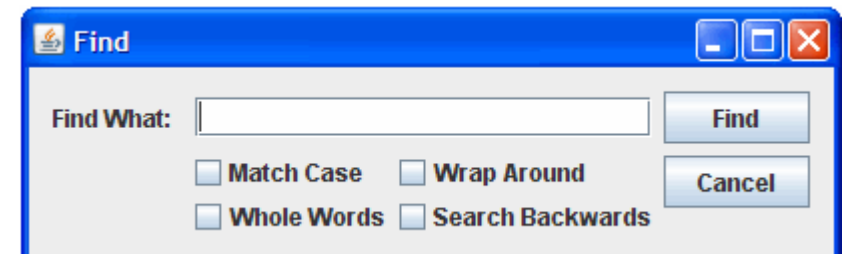
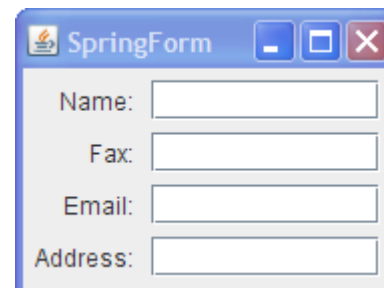
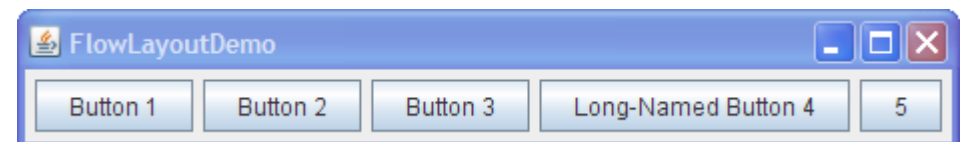
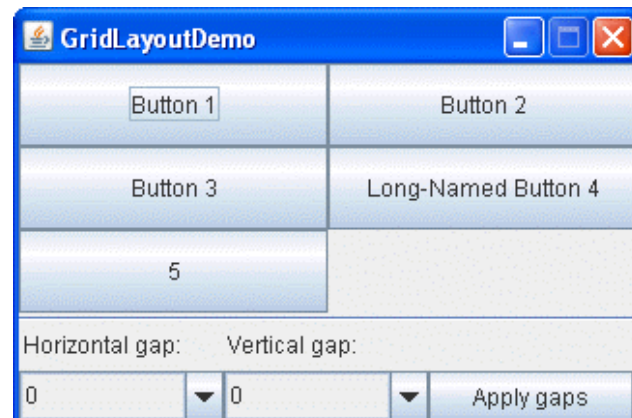
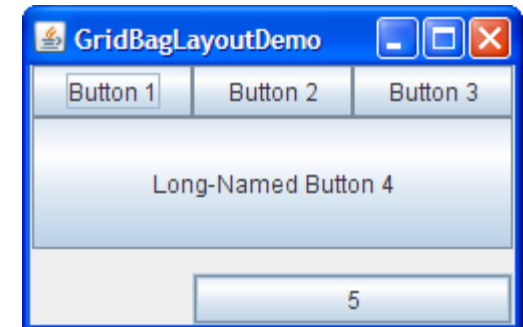
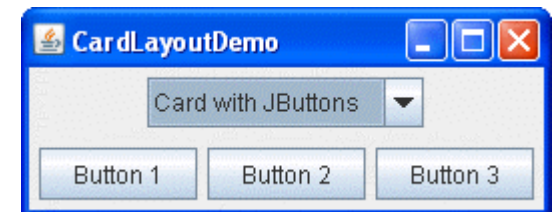
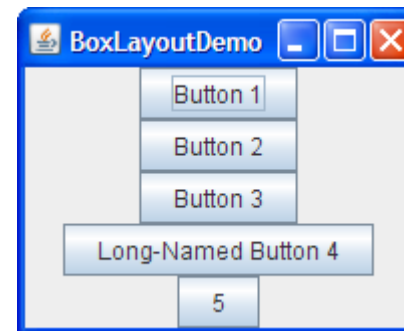
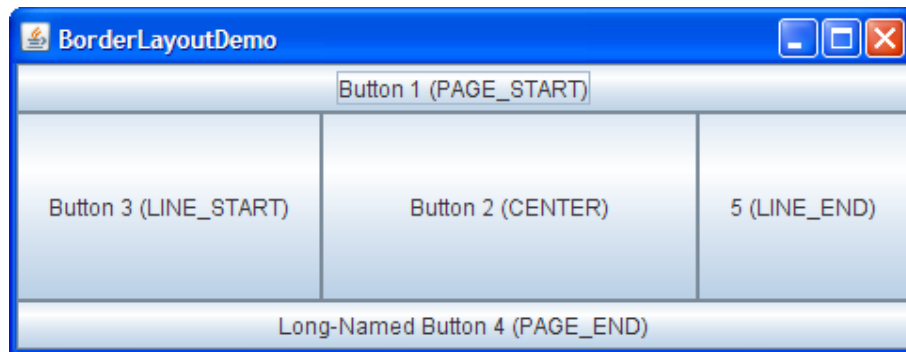
dynamique



GL Avancé: Prototypage et interfaces utilisateur

Disposition des composants (API Swing)

- Mise en page dynamique: choix du layout (Swing)

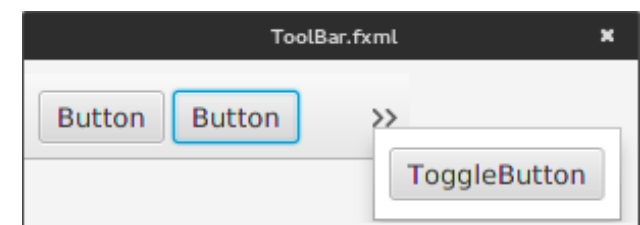
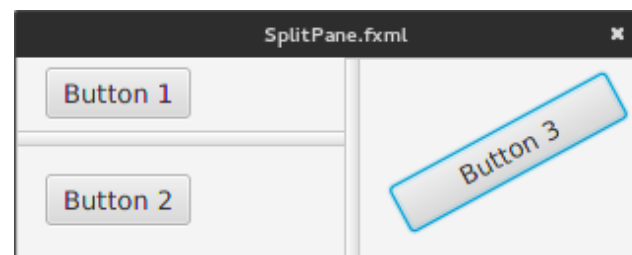
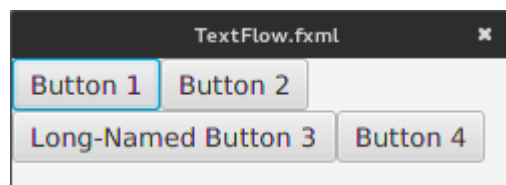
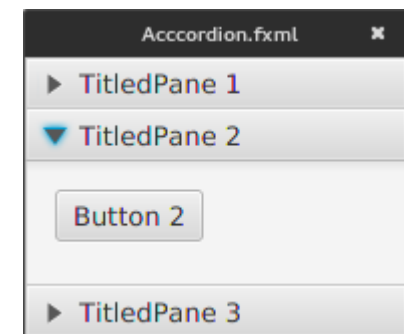
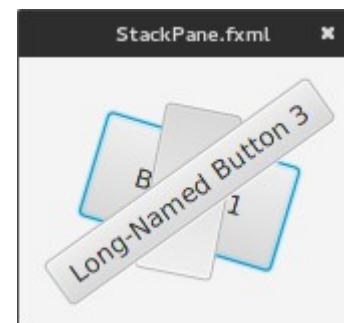
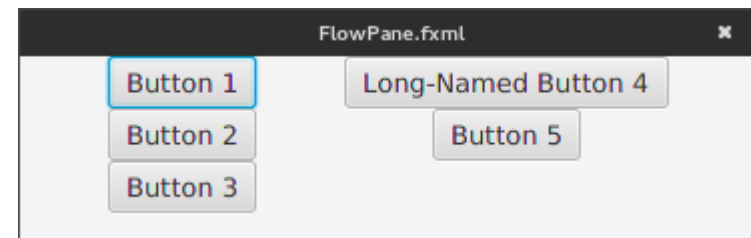
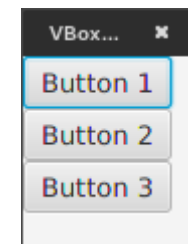
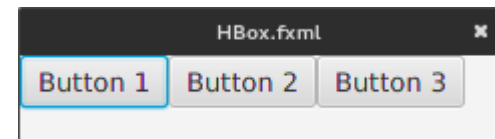
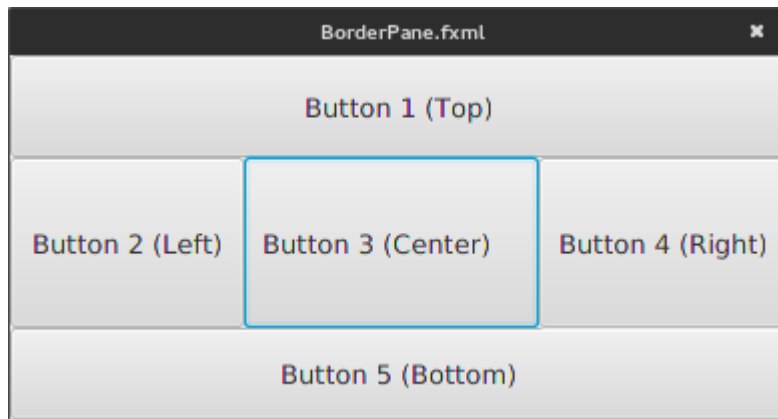


GL Avancé: Prototypage et interfaces utilisateur

20 / 51

Disposition des composants (API JavaFX)

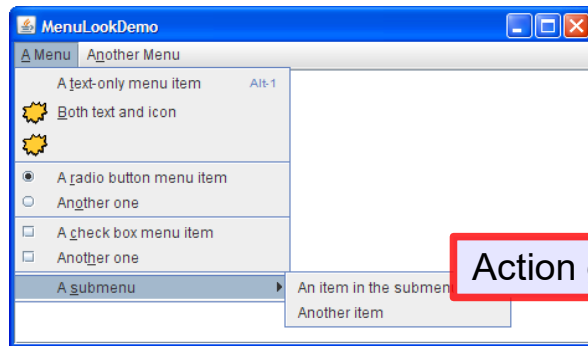
- Mise en page dynamique: choix du pane (JavaFX)



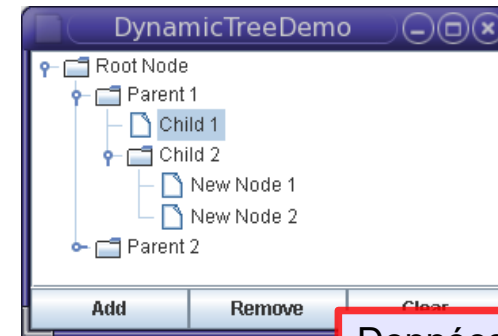
GL Avancé: Prototypage et interfaces utilisateur

Traduction des besoins

- Assemblage d'éléments d'interface utilisateur: Menu, tableau, liste, arbre, checkbox, ...
- Écrans, boîtes de dialogue, viewers, diagrammes, composants métiers ...



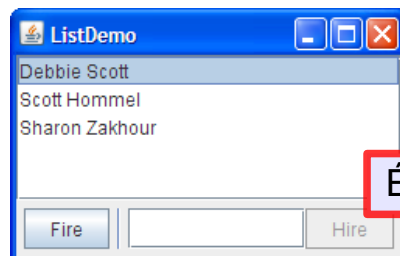
Action globale



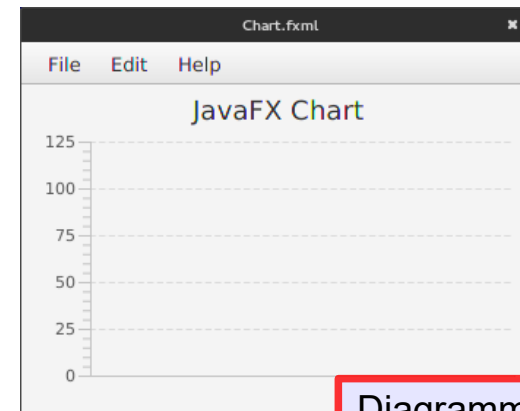
Données hiérarchiques

First Name	Favorite Color	Sport	# of Years	Vegetarian
Mary	Purple	Snowboarding	5	<input type="checkbox"/>
Alison	Blue	Rowing	3	<input checked="" type="checkbox"/>
Kathy	Green	Kayaking	2	<input type="checkbox"/>
Sharon	Red	Swimming	20	<input checked="" type="checkbox"/>

Matrice bi-dimensionnelle



Énumération



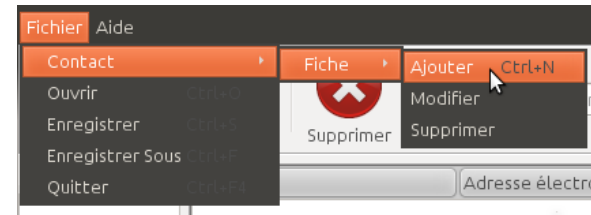
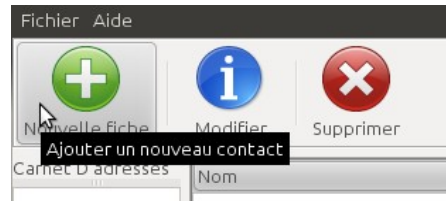
Diagrammes

GL Avancé: Prototypage et interfaces utilisateur

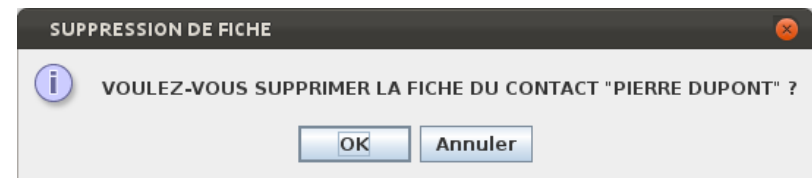
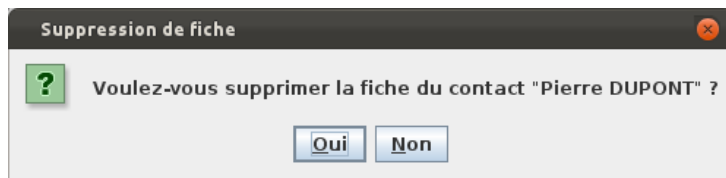
22 / 51

11 règles d'ergonomie

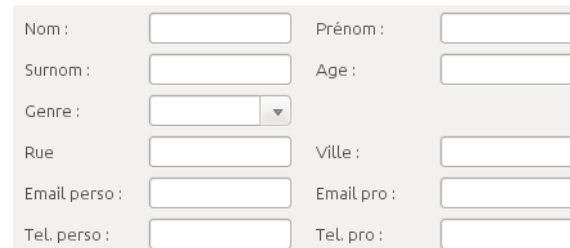
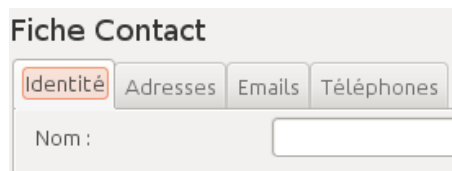
1. Incitation : orienter l'utilisateur de façon intuitive



2. Lisibilité : faciliter la lecture et la compréhension



3. Groupement : organiser logiquement les éléments pour mieux les distinguer



4. Feedback direct : informer de ce qui se passe



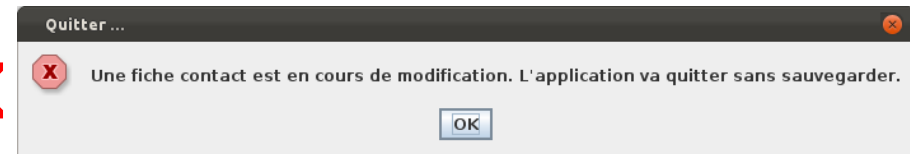
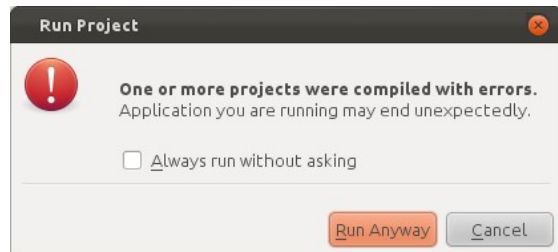
Blocage de l'interface pendant le traitement

GL Avancé: Prototypage et interfaces utilisateur

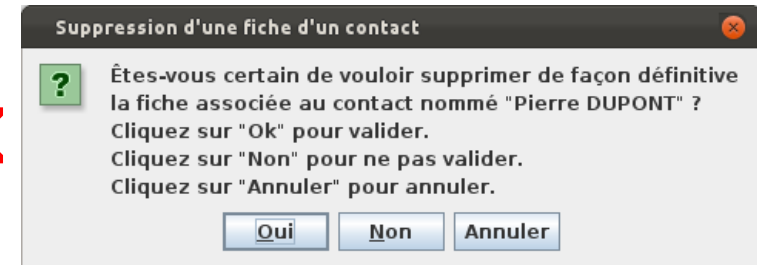
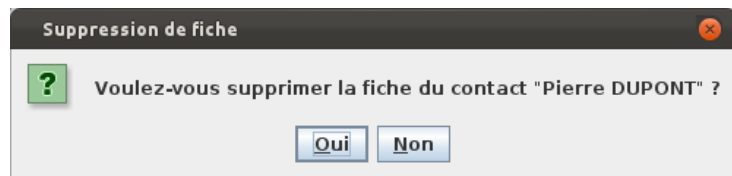
23 / 51

11 règles d'ergonomie

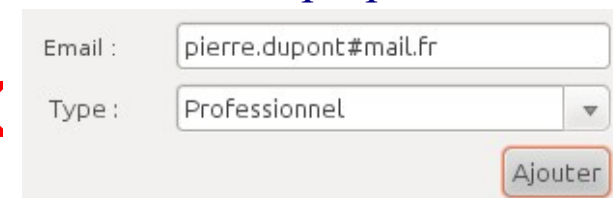
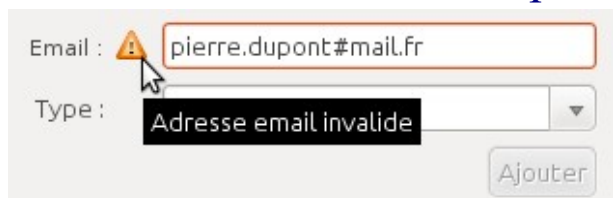
5. Contrôle utilisateur : laisser le contrôle des actions du système



6. Concision : brièveté des messages pour aller à l'essentiel et réduire la charge de travail



7. Gestion des erreurs : récupérer les erreurs, notifier l'utilisateur, proposer des solutions



GL Avancé: Prototypage et interfaces utilisateur

24 / 51

11 règles d'ergonomie

8. Adaptabilité : proposer plusieurs mécanismes d'utilisation pour s'adapter à l'utilisateur

- déclencher une même action de différentes façons (bouton, raccourci clavier, menu contextuel, ...)
- préférences utilisateurs (colonnes des tableaux configurables, look and feel personnalisable, ...)
- support du glisser / déplacer

9. Cohérence : assurer une homogénéité contextuelle, stabiliser les choix ergonomiques

Nom : Prénom :

Genre : ☐ Homme ☐ Femme

Email :

Type : ☐ Professionnel ☐ Personnel



Nom : Prénom :

Genre :

Email :

Type : ☐ Professionnel ☐ Personnel

10. Signifiante : utiliser un langage explicite en adéquation avec la finalité du message



11. Compatibilité :

- s'accorder avec le mode opératoire de l'utilisateur
- respecter les règles d'ergonomie usuelles des applications similaires

GL Avancé: Prototypage et interfaces utilisateur

25 / 51

Évaluation

- **Critères d'utilité :**
 - Réponse aux besoins spécifiés
 - Pertinence de la réponse
 - **Critères d'utilisabilité :**
 - Efficacité : atteindre le résultat prévu
 - Efficience : atteindre le résultat avec un effort moindre
 - Fiabilité : résultat obtenu sans erreur ou erreur correctement gérée
 - Satisfaction : évaluation subjective de l'interaction
 - **Méthodes d'évaluation :**
 - Par l'utilisateur : panel de testeur, enquête satisfaction, expérimentation
 - Par l'expertise : audit ergonomique, évaluation heuristique (Nielsen), prototypage
- **Valider la conception de l'interface homme-machine**

GL Avancé: Prototypage et interfaces utilisateur

26 / 51

Compléments

- **La charte graphique :**

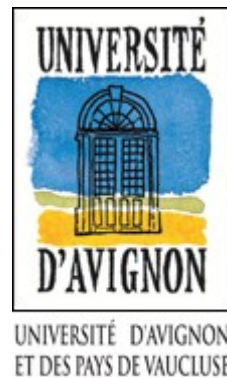
- Description des éléments d'interface (Swing)
- Police, taille et couleur des caractères
- Cote en pixel des écrans
- Code des couleurs
- Bibliothèque d'icônes

- **Le story bord :**

- Scenarii d'utilisation
- Cinématique de navigation
- Enchaînement des différents écrans

→ **Consolider l'identité visuelle et le design de l'interface homme-machine**

Prototypage en Swing avec NetBeans



GL Avancé: Prototypage et interfaces utilisateur

28 / 51

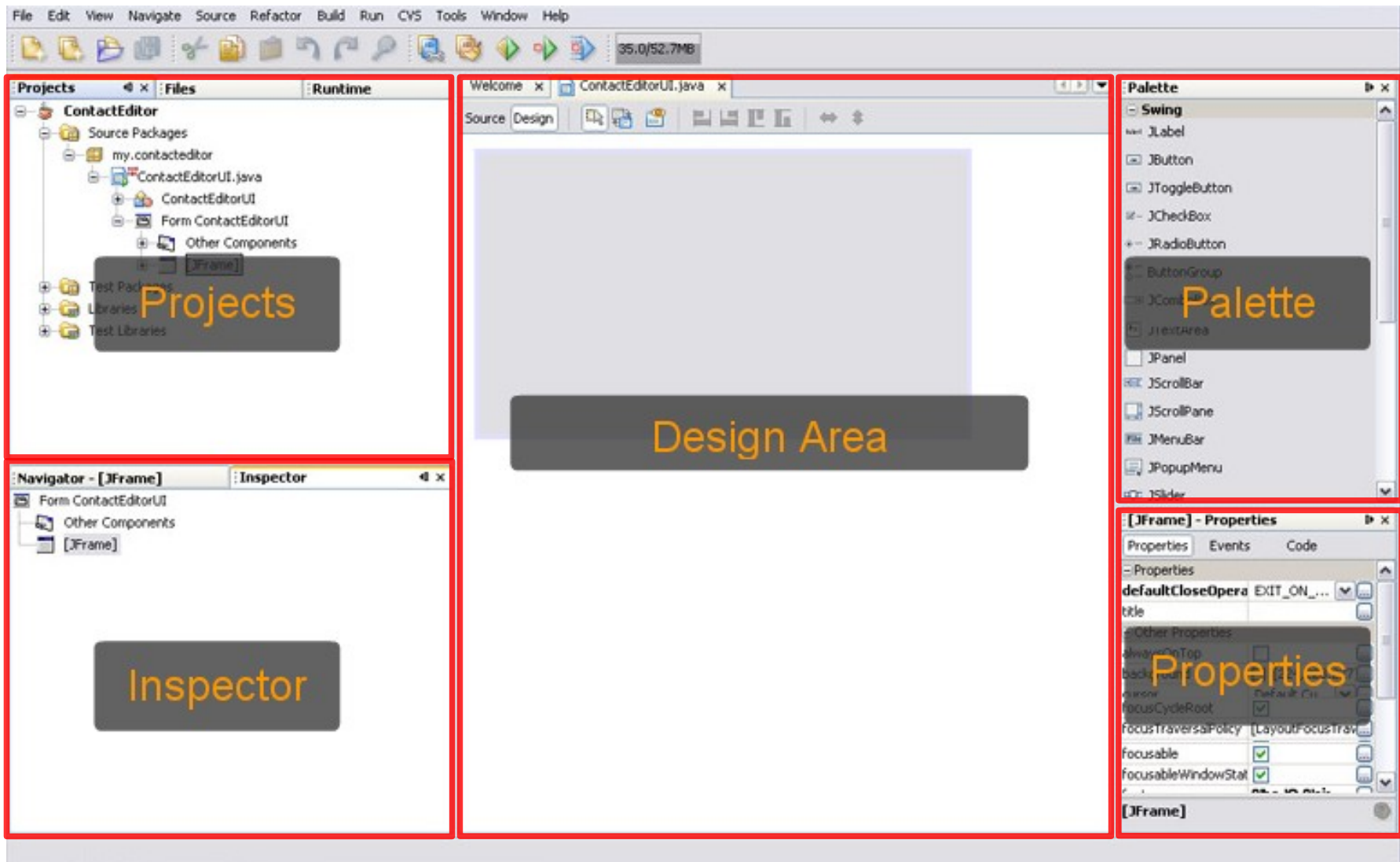
Prototypage Swing avec NetBeans

- **Points forts de l'IDE NetBeans :**
 - Moteur d'affichage de composants graphiques
 - Designer WYSIWYG par simple glisser/déplacer depuis la palette
 - Éditeur des propriétés des composants Swing
 - Génération du code Java correspondant au prototypage
 - Insertion de code pour enrichir le prototypage
 - Création de composants/modules réutilisables (Bean)
 - Système de palette personnalisable
 - Intégration du Beans Binding avec conversion et validation
 - Matisse: layout manager intuitif pour le prototypage simple rapide
 - Débogueur visuel de l'interface utilisateur

GL Avancé: Prototypage et interfaces utilisateur

29 / 51

Prototypage Swing avec NetBeans

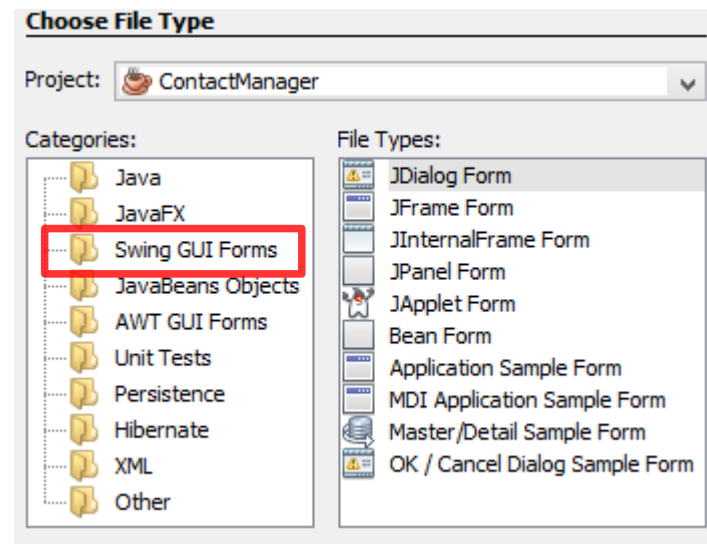


GL Avancé: Prototypage et interfaces utilisateur

30 / 51

Prototypage Swing avec NetBeans

- **Projet de type « Java Application »**
 - Ajout d'écrans depuis la catégorie « Swing GUI Forms » :
 - ✓ JDialog Form: boîte de dialogue
 - ✓ JFrame Form: fenêtre principale
 - ✓ JPanel Form: conteneur de composants
 - ✓ Bean Form : formulaire basé sur un composant JavaBeans (ex : un autre formulaire)

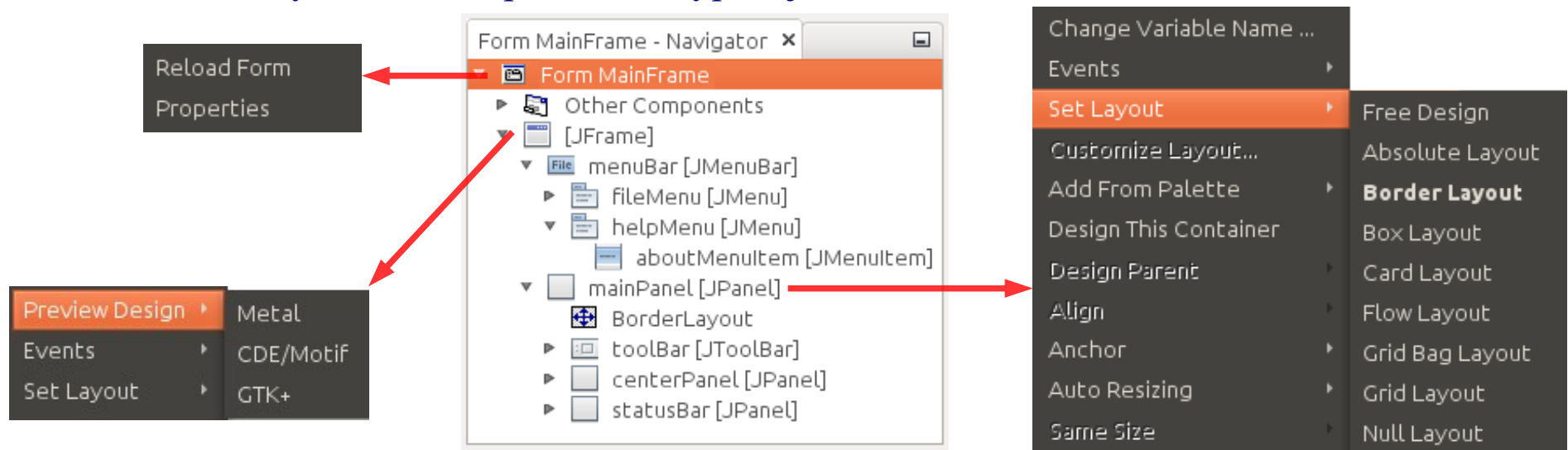


GL Avancé: Prototypage et interfaces utilisateur

31 / 51

Prototypage Swing avec NetBeans

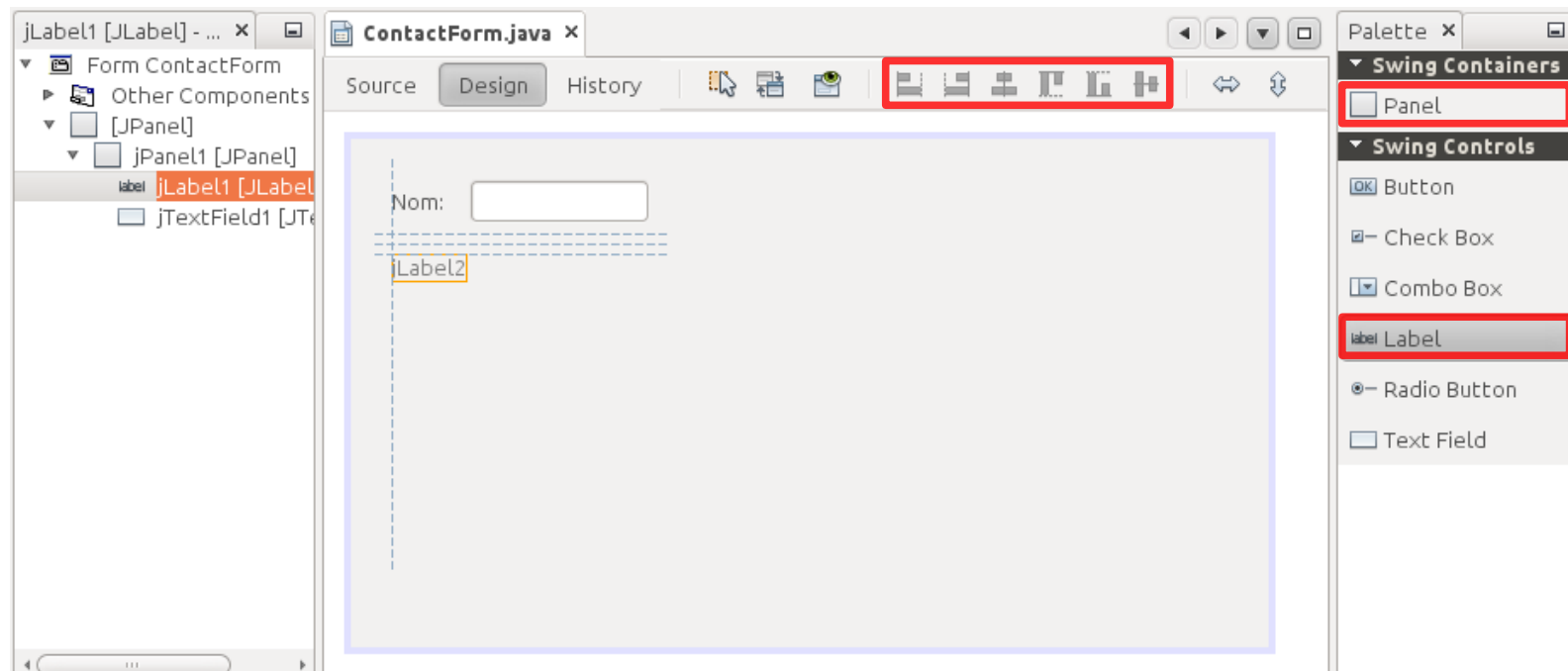
- **La vue « Inspector » (ou « Navigator »)**
 - Vue arborescente de l'imbrication des composants
 - Déplacement de composants par glisser/déplacer
 - Édition du nom des variables associées aux composants
 - Prévisualisation des composants de type « java.awt.Window » avec choix du L&F
 - Rechargement de la maquette : action « Reload Form » sur la racine
 - Choix du layout des composants de type « java.awt.Container »



GL Avancé: Prototypage et interfaces utilisateur

Prototypage Swing avec NetBeans

- **Les vues « Design » et « Palette » : Swing GUI Builder**
 - Disposition de conteneurs dans une zone représentant la maquette
 - Insertion de widgets
 - Édition du texte affiché
 - Alignement et ancrage (selon le layout choisi, par défaut « Free Design »)

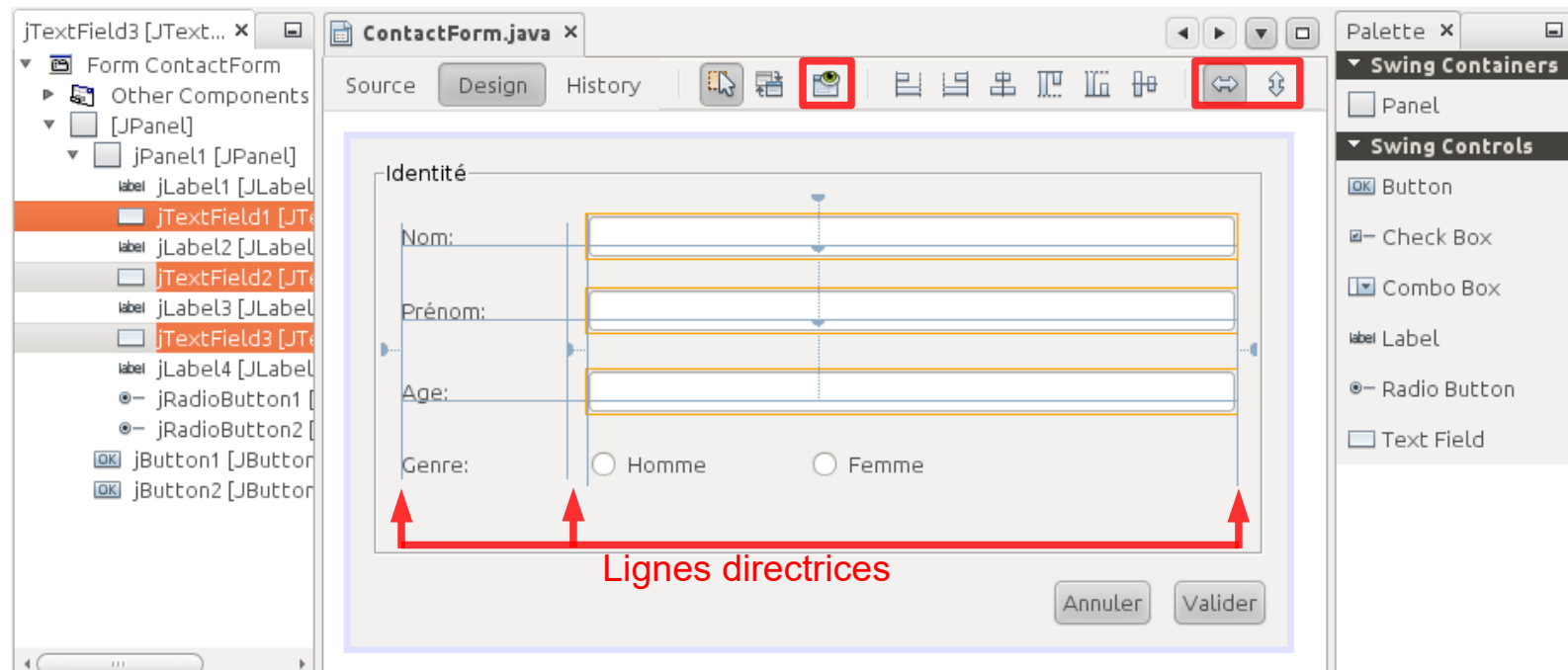
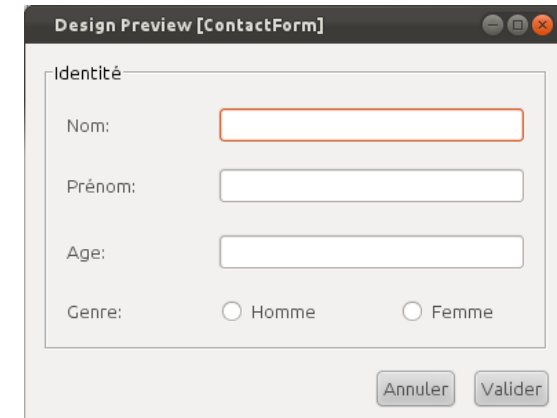


GL Avancé: Prototypage et interfaces utilisateur

33 / 51

Prototypage Swing avec NetBeans

- Redimensionnement contextuel
- Indentation assistée
- Ajustement horizontal/vertical automatique
- Prévisualisation du rendu



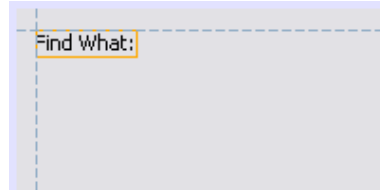
GL Avancé: Prototypage et interfaces utilisateur

34 / 51

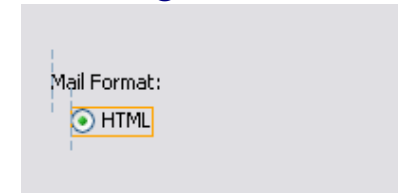
Prototypage Swing avec NetBeans

➤ Légende des indications visuelles d'alignement

✓ Inset : espace entre composant et conteneur



✓ Indentation : relation d'alignement décalé



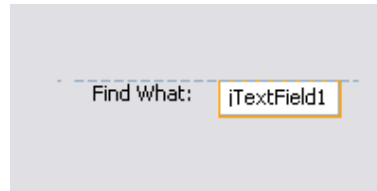
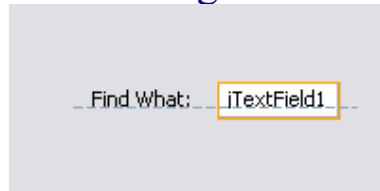
✓ Offset : espace entre les composants adjacents



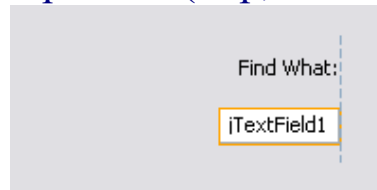
✓ Preferred Distance : taille de l'intervalle



✓ Baseline : relation d'alignement du texte



✓ Edge : relation d'alignement des composants (Top, Bottom, Left, Right)

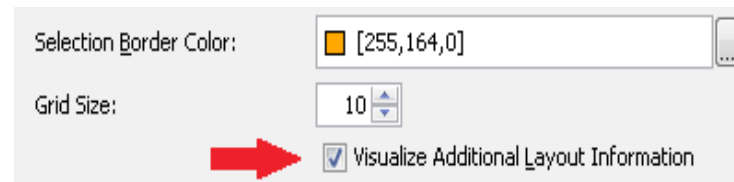


GL Avancé: Prototypage et interfaces utilisateur

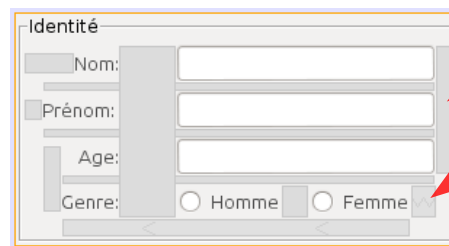
35 / 51

Prototypage Swing avec NetBeans

- Visualisation des intervalles (NetBeans 7.2+)
 - ✓ Configuration depuis le menu Tools > Options > Java > GUI Builder

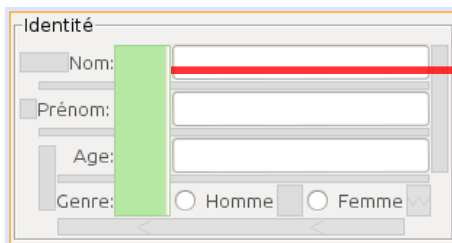


- ✓ Affichage lors de la sélection du conteneur (caché lors de la désélection)

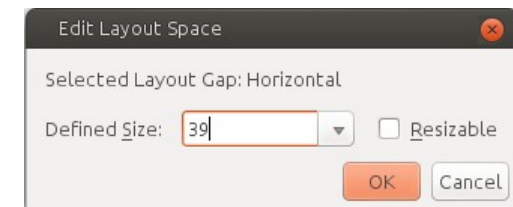
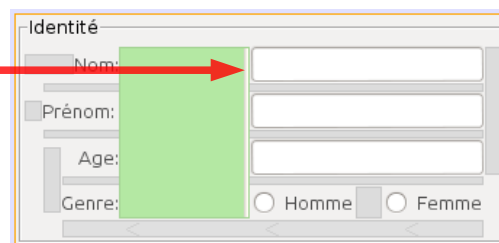


- Manipulation des intervalles (NetBeans 7.2+)

- ✓ Édition à la souris, par double clic ou par le menu contextuel « Edit Layout Space »



Resize



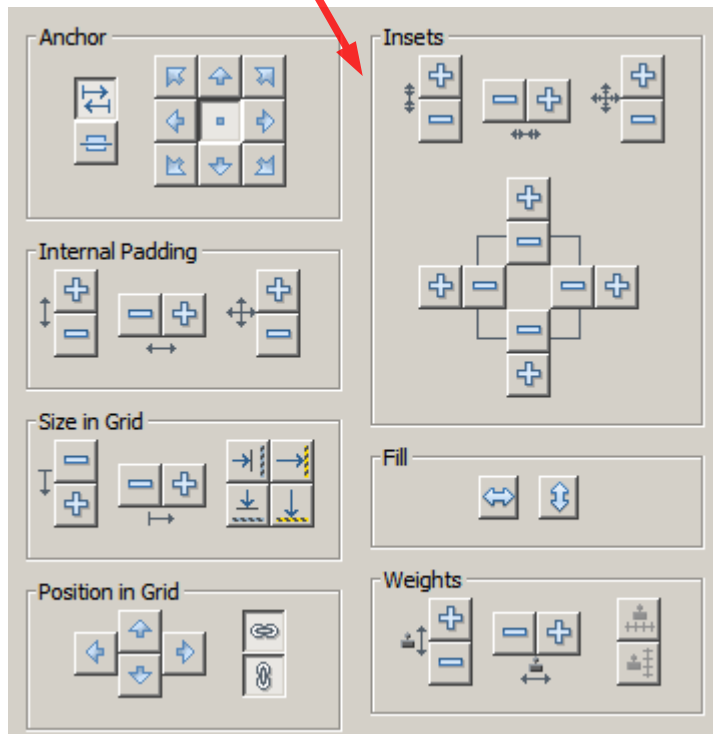
GL Avancé: Prototypage et interfaces utilisateur

Prototypage Swing avec NetBeans

• Personnalisation du « GridBagLayout »

- Layout manager flexible et complexe sous forme de grille de lignes et colonnes
- Application de contraintes sur les cellules : alignement, espacement, remplissage, etc...

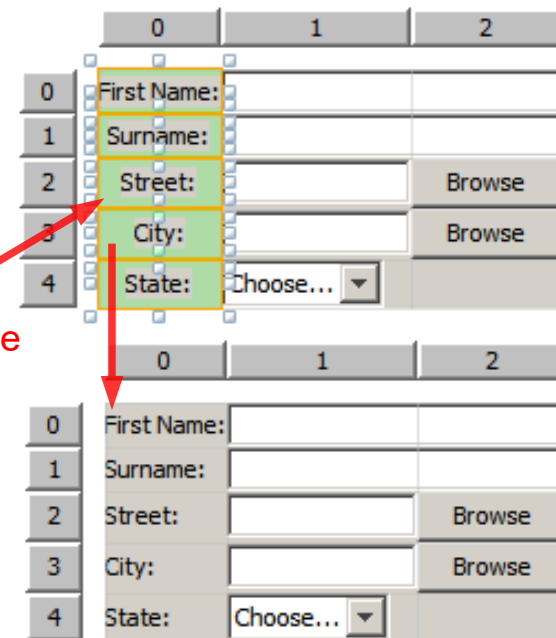
Property customizer



Barre d'outil spécifique



Ex : ancrage à gauche



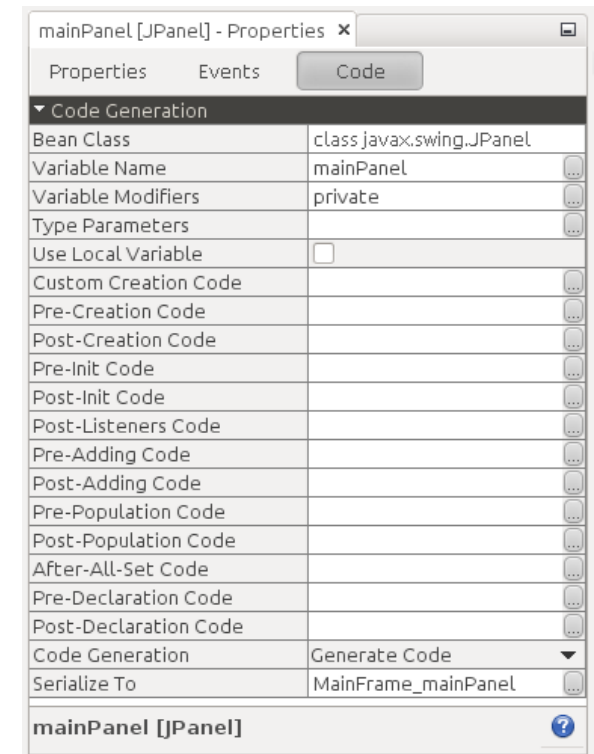
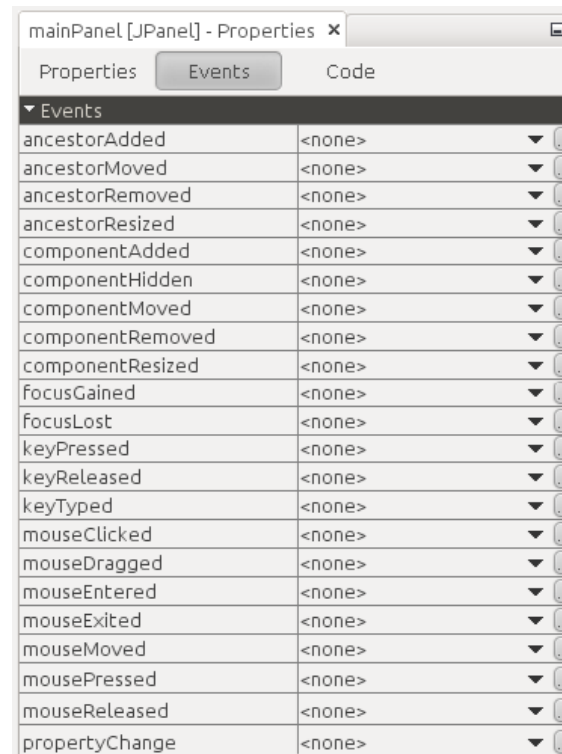
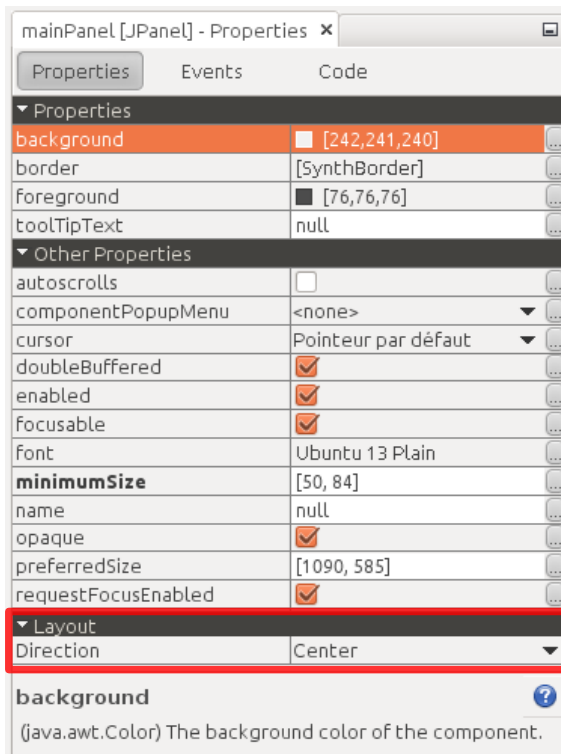
GL Avancé: Prototypage et interfaces utilisateur

37 / 51

Prototypage Swing avec NetBeans

• La vue « Properties »

- ✓ Accès aux propriétés du composant (couleur, bordure, taille, ...)
- ✓ Configuration de la disposition en fonction du layout du conteneur parent
- ✓ Gestion des événements liés aux écouteurs enregistrables sur le composant
- ✓ Affinage du code généré avec possibilité d'insérer du code manuellement



GL Avancé: Prototypage et interfaces utilisateur

38 / 51

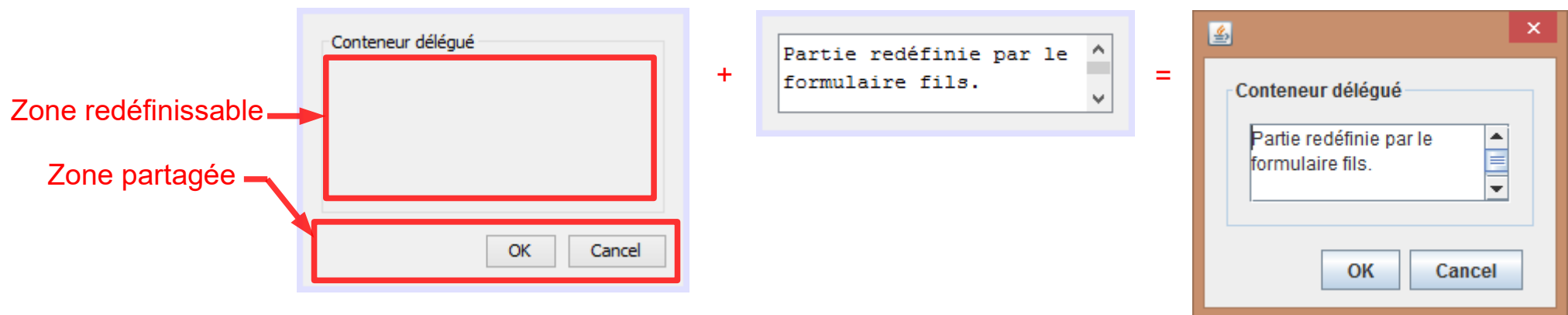
Prototypage Swing avec NetBeans

- **Composition de formulaires**

- Un formulaire peut être inséré dans un autre formulaire par simple glisser/déplacer
- Un formulaire peut être stocké sous forme de bean dans la palette et être réutilisé ailleurs

- **Extension de formulaire**

- Un formulaire peut servir de modèle
- S'il n'est pas vide, nécessité de déclarer un conteneur délégué vide
- Il doit être un JavaBeans : constructeur par défaut et getters/setters
- Exemple : création d'un formulaire pour une boîte de dialogue OK/Cancel avec texte libre



GL Avancé: Prototypage et interfaces utilisateur

39 / 51

Prototypage Swing avec NetBeans

- **Déclaration du conteneur délégué**

- Ajouter un bean info au formulaire modèle : menu contextuel → « BeanInfo Editor ... »
- Dans la classe bean info, déclarer le nom de la méthode qui retourne le conteneur délégué

```
private static BeanDescriptor getBdescriptor()
{
    BeanDescriptor beanDescriptor = new BeanDescriptor (m1.pi.OkCancelDialog.class , null);

    // Here you can add code for customizing the BeanDescriptor.
    beanDescriptor.setValue("containerDelegate", "getDelegateContainer");

    return beanDescriptor;
}
```

- **Constructeur par défaut du formulaire modèle :**

- Appeler l'initialisation du formulaire
- Form Size Policy : « generate pack() »



```
public OkCancelDialog()
{
    initComponents();
}
```

GL Avancé: Prototypage et interfaces utilisateur

40 / 51

Prototypage avec NetBeans

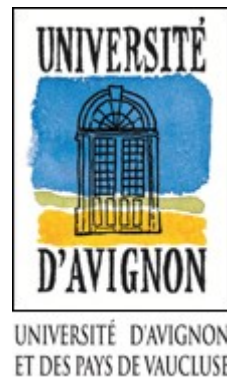
- **Contrôle du rendu réel**

- Choix du projet principal : Menu "Run" -> "Set Main Project"
- Compiler l'ensemble des classes :  ou F9
- Exécution du projet :  ou F6 (ou Maj+F6 pour exécuter le main du fichier sélectionné)
- Forcer le même L&F que celui utilisé pour la prévisualisation :

```
try
{
    // Set System L&F before displaying main frame
    UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
}
catch (UnsupportedLookAndFeelException e) { }
catch (ClassNotFoundException e) { }
catch (InstantiationException e) { }
catch (IllegalAccessException e) { }

new MainFrame().setVisible(true);
```

Prototypage en JavaFX / FXML avec NetBeans et Scene Builder



GL Avancé: Prototypage et interfaces utilisateur

42 / 51

Prototypage JavaFX avec NetBeans et Scene Builder

- **Points forts de l'IDE NetBeans :**
 - Insertion de code pour enrichir le prototypage
 - Intégration complète avec Scene Builder
 - Débogueur visuel de l'interface utilisateur
- **Points forts de Scene Builder**
 - Moteur d'affichage de composants graphiques
 - Layout manager intuitif pour le prototypage simple, rapide, et dynamique
 - Designer WYSIWYG par simple glisser/déplacer depuis la palette
 - Éditeur des propriétés des composants JavaFX
 - Création de composants/modules réutilisables (Custom FXML/Jar import)
 - Génération du code FXML correspondant à la structure de l'interface utilisateur
 - Analyseur CSS intégré pour la personnalisation du rendu de l'interface

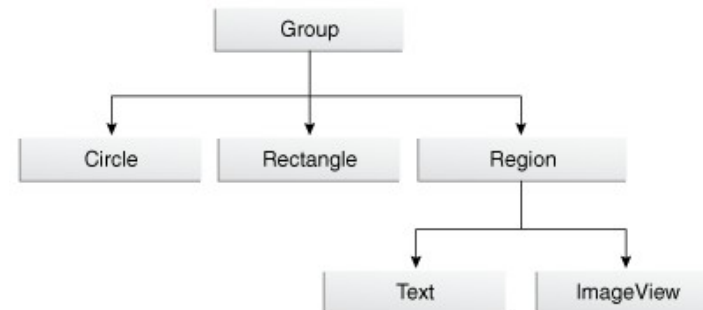
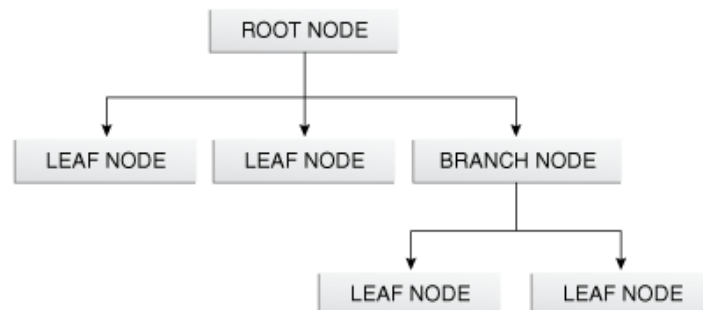
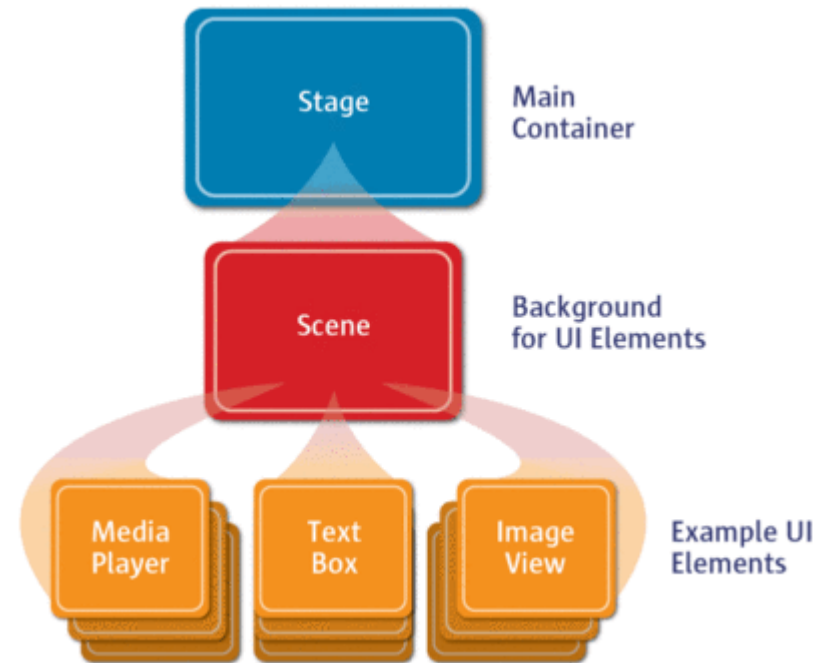
GL Avancé: Prototypage et interfaces utilisateur

43 / 51

JavaFX architecture

- **Structure d'une interface JavaFX :**

- **Stage** : conteneur principal de la fenêtre applicative
- **Scene** : structure hiérarchique de nœuds contenant les éléments graphiques
- **Node** : les composants graphiques :
 - ✓ les formes géométriques (Shapes : Arc, Circle, Box, ...)
 - ✓ les contrôles (Button, CheckBox, Label, ...)
 - ✓ les panneaux (Containers : Pane, BorderPane, ...)
 - ✓ Des graphiques (AreaChart, BarChart, PieChart, ...)
 - ✓ ou encore des objets 3D (PointLight, ParallelCamera, ...)

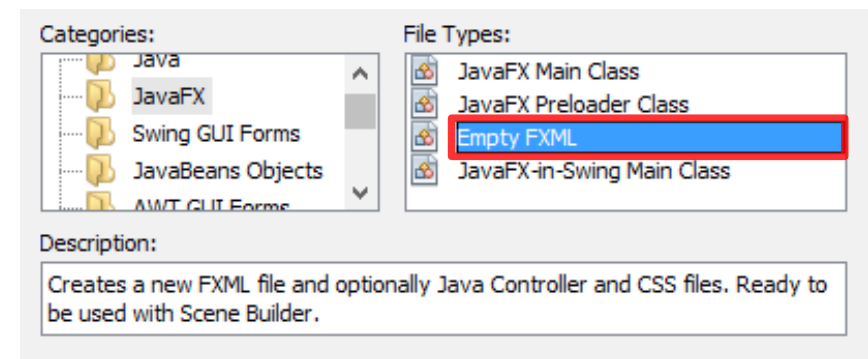
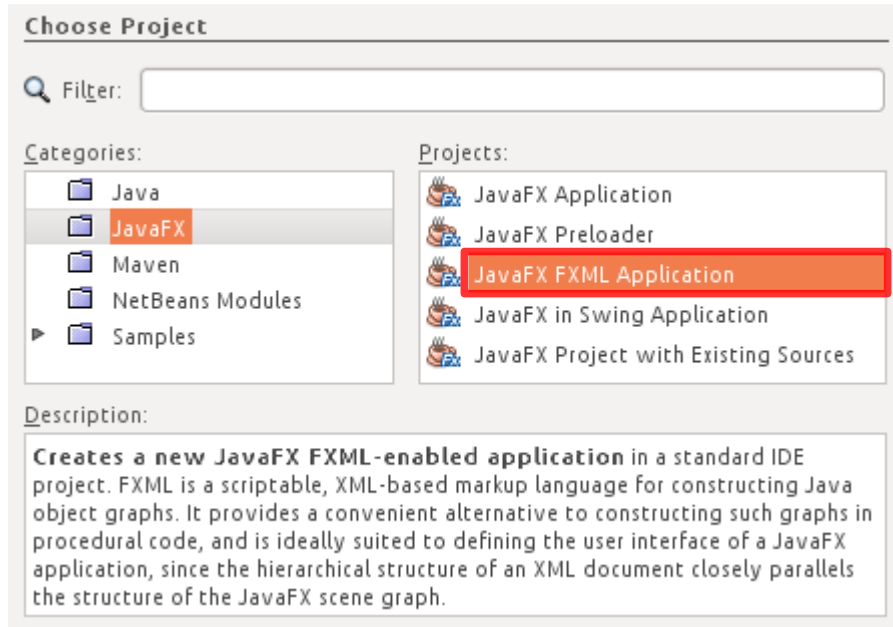


GL Avancé: Prototypage et interfaces utilisateur

44 / 51

Prototypage JavaFX avec NetBeans

- **Contenu de type « JavaFX »**
 - Nouveau projet dans la catégorie « JavaFX » :
 - ✓ JavaFX FXML Application: Fenêtre principale avec descripteur d'interface FXML



- Nouveau fichier dans la catégorie « JavaFX » :
 - ✓ Empty FXML : composant JavaFX avec controler Java et CSS

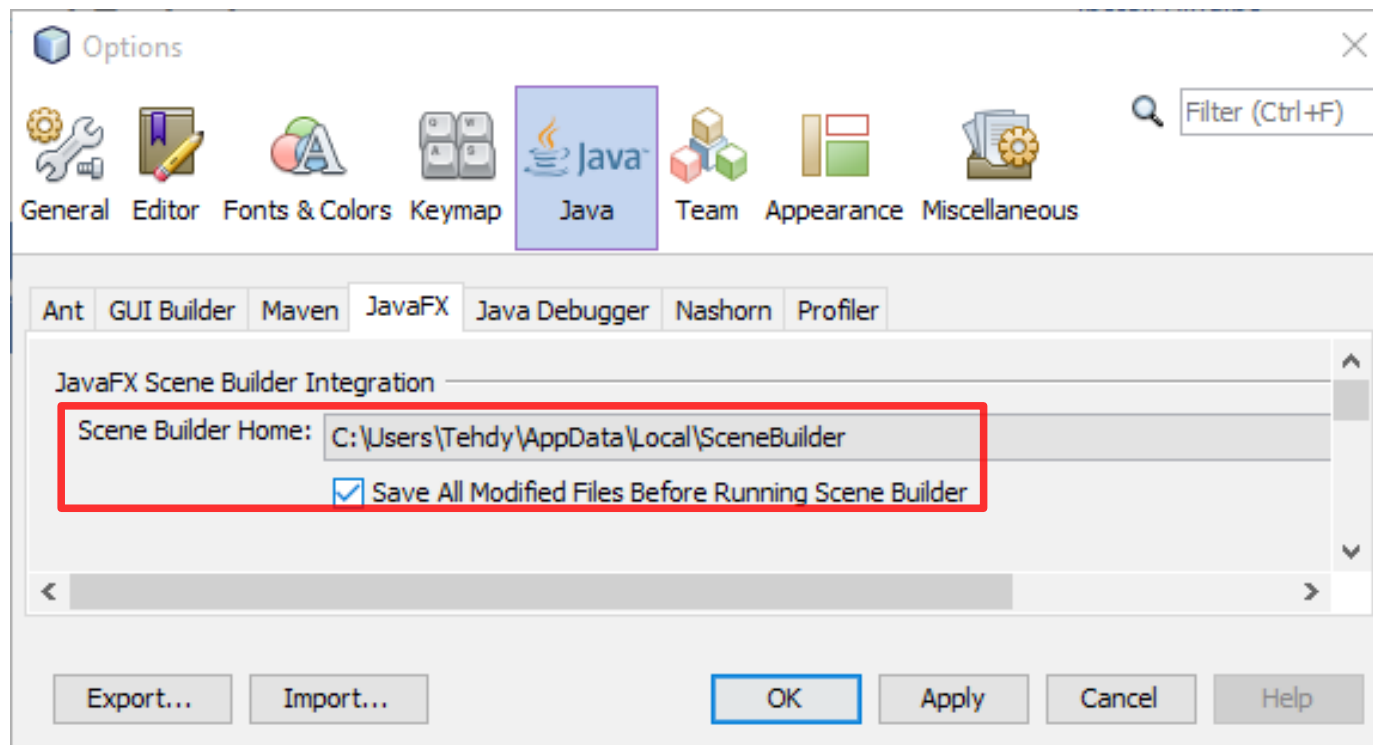
GL Avancé: Prototypage et interfaces utilisateur

45 / 51

Prototypage JavaFX avec NetBeans

- **Intégration de Scene Builder à NetBeans**

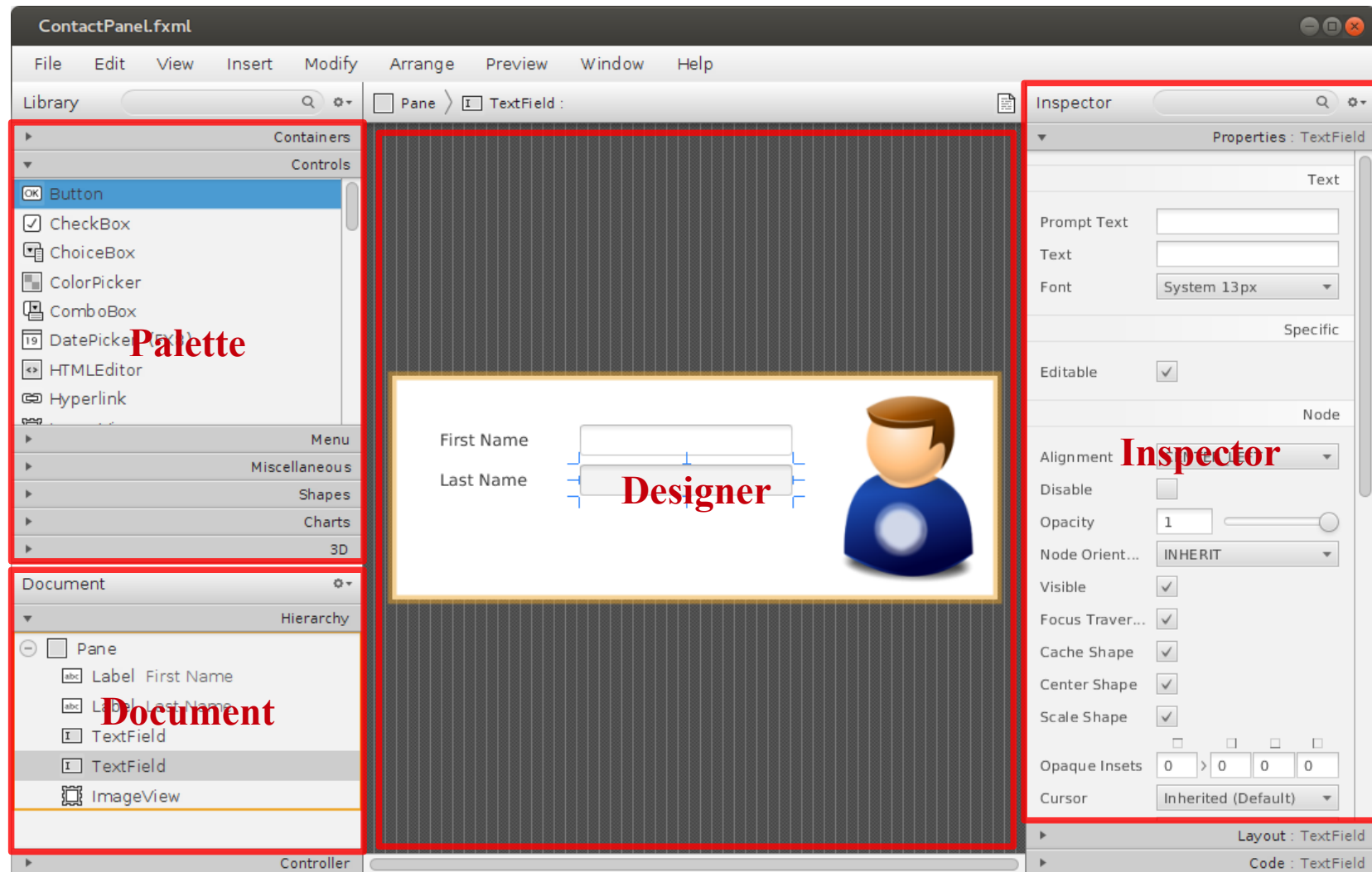
- Depuis le menu principal, « Tools » -> « Options » -> « Java » :
 - ✓ Sélectionner le répertoire d'installation de Scene Builder
 - ✓ Cocher la case pour sauver dans NetBeans les fichiers avant de les envoyer à Scene Builder



GL Avancé: Prototypage et interfaces utilisateur

JavaFX Scene Builder

- Outil de prototypage : Scene Builder 8.x



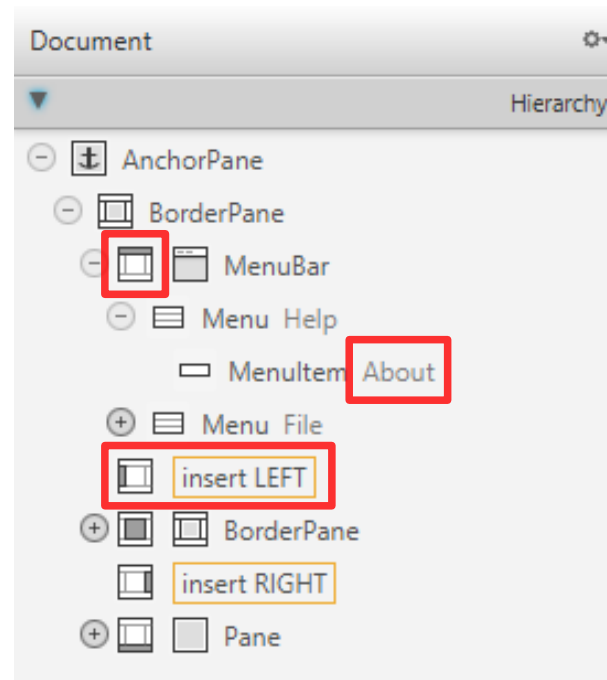
GL Avancé: Prototypage et interfaces utilisateur

47 / 51

Prototypage JavaFX avec Scene Builder

- **La vue « Document »**

- Vue arborescente de l'imbrication des composants
- Déplacement de composants par glisser/déplacer
- Édition de la valeur des composants affichant du texte
- Affichage du layout des containers et de la position des controls
- Indication des nœuds fils compatibles

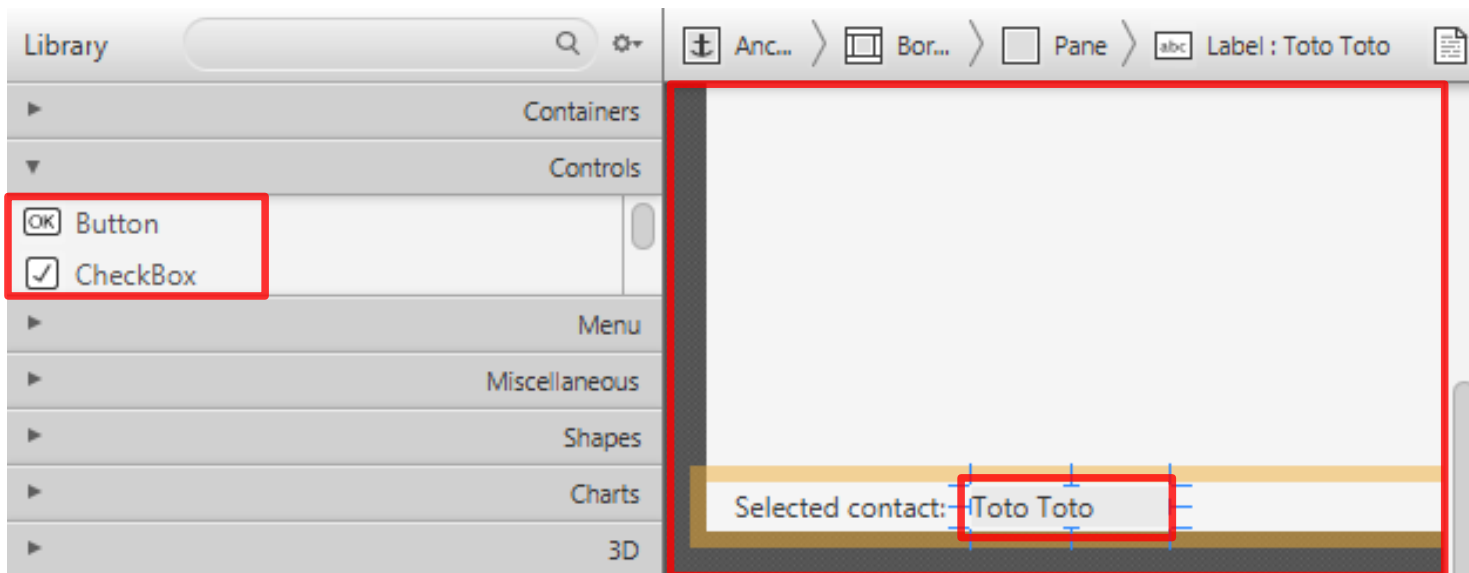


GL Avancé: Prototypage et interfaces utilisateur

48 / 51

Prototypage JavaFX avec Scene Builder

- **Les vues « Design » et « Palette » : JavaFX GUI Builder**
 - Disposition de composants dans une zone représentant la maquette
 - Insertion de widgets depuis la palette (ou depuis le menu contextuel)
 - Édition du texte affiché
 - Assistant d'alignement et d'ancrage (uniquement pendant le déplacement)



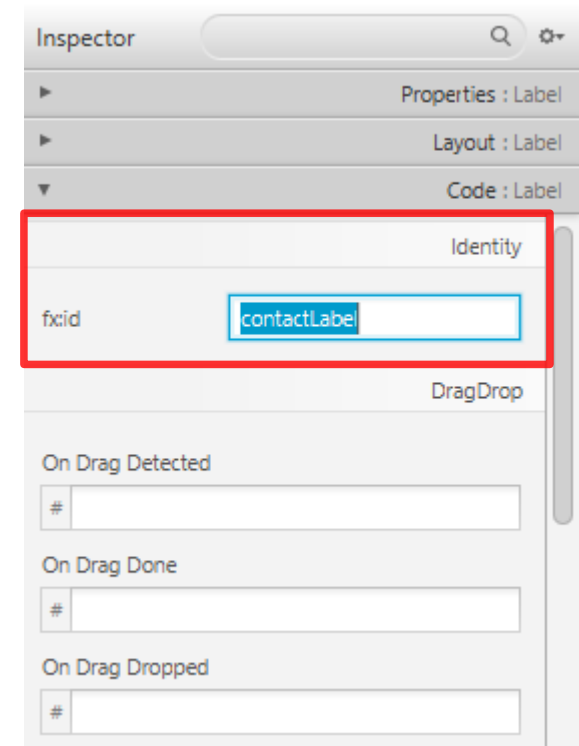
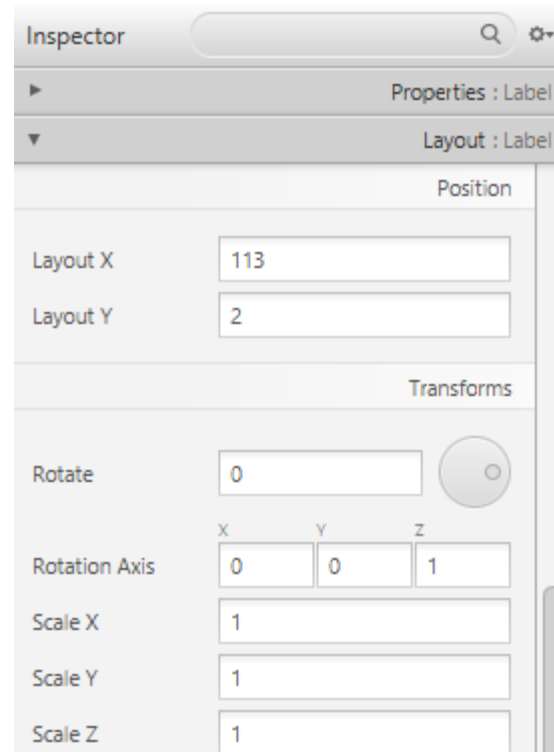
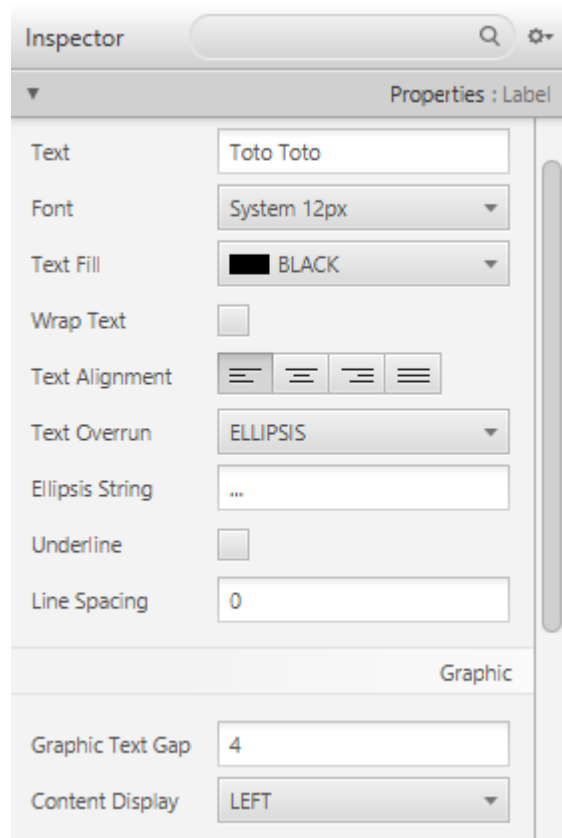
GL Avancé: Prototypage et interfaces utilisateur

49 / 51

Prototypage JavaFX avec Scene Builder

• La vue «Inspector»

- ✓ Accès aux propriétés du composant (couleur, alignement, styles CSS, ...)
- ✓ Mise en forme du composant (position, taille, transformation, ...)
- ✓ Code java associé (dont fx:id) et événements liés aux écouteurs enregistrables sur le composant



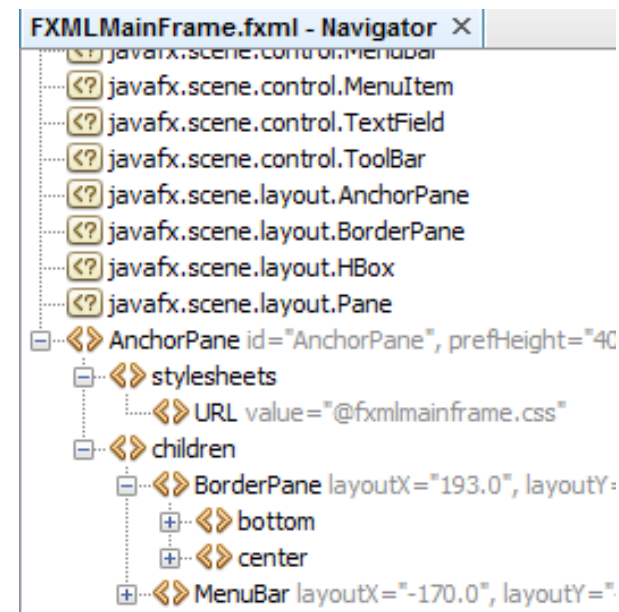
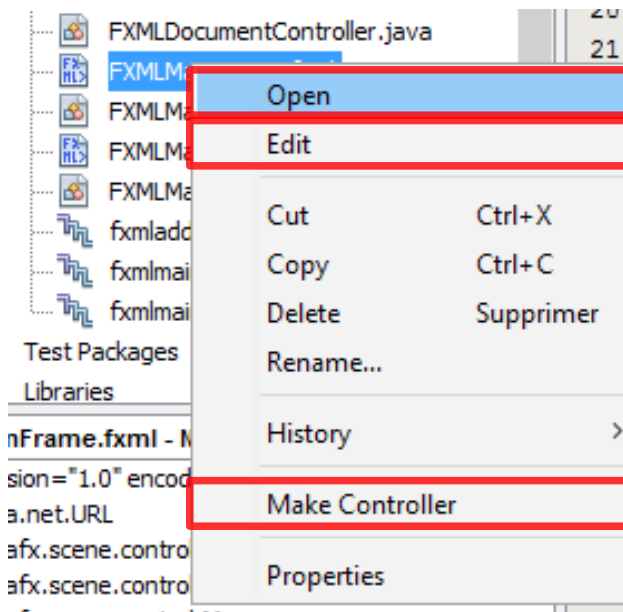
GL Avancé: Prototypage et interfaces utilisateur

50 / 51

Prototypage JavaFX avec NetBeans

• Synchronisation entre NetBeans et Scene Builder

- ✓ Depuis un fichier FXML, double click ou menu « Open » pour ouvrir dans Scene Builder
- ✓ Depuis un fichier FXML, menu contextuel « Edit » pour ouvrir le fichier au format XML
- ✓ La mise à jour du controller Java se fait manuellement par le menu contextuel « Make Controller »
- ✓ La vue « Navigator » affiche la hiérarchie de l'objet FXML



2ème partie:

Prototypage fonctionnel

