🎨Guide complet de FlatLaf pour Swing

FlatLaf est une bibliothèque moderne de Look and Feel (L&F) pour **Java Swing**, inspirée de **Material Design** et **IntelliJ IDEA**. Il rend vos interfaces plus élégantes, cohérentes et modernes — sans changer votre code Swing existant.

🧩1. Installation

Étape 1 : Téléchargement

Télécharge le fichier JAR depuis :
👉https://www.formdev.com/flatlaf/

Exemple : `flatlaf-3.4.1.jar`

Étape 2 : Placement dans ton projet

Place le fichier dans ton dossier :

```
src/lib/flatlaf-3.4.1.jar
```

Étape 3 : Compilation et exécution

Dans ton script Bash :

```bash
#!/bin/bash

javac -cp src/lib/ojdbc17.jar:src/lib/designgridlayout-1.6.jar:src/lib/
flatlaf-3.4.1.jar -d bin $(find . -name "*.java") -Xlint:unchecked

if [ $? -ne 0 ]; then
    echo "\nCompilation échouée ! Exécution annulée."
    exit 1
fi

echo "Compilation terminée avec succès !"
echo "Exécution ..."
echo "----------------------------------------"

java -cp bin:src/lib/ojdbc17.jar:src/lib/designgridlayout-1.6.jar:src/lib/
flatlaf-3.4.1.jar goldman.main.Main
```

---

2. Activation du Look & Feel

Ajoute simplement au début de ton programme :

```
import com.formdev.flatlaf.FlatLightLaf;

public class Main {
    public static void main(String[] args) {
        FlatLightLaf.setup(); // Active le thème clair

        javax.swing.SwingUtilities.invokeLater(() -> {
            new FenetrePrincipale();
        });
    }
}
```

🎨3. Les thèmes disponibles

FlatLaf propose plusieurs styles par défaut :

```
import com.formdev.flatlaf.*;

FlatLightLaf.setup();     // Thème clair
FlatDarkLaf.setup();      // Thème sombre
FlatIntelliJLaf.setup();  // Style IntelliJ clair
FlatDarculaLaf.setup();   // Style IntelliJ sombre
```

Tu peux changer le thème à tout moment avant d'afficher la fenêtre principale.

---

4. Personnalisation globale

Tu peux ajuster les propriétés UI via `UIManager` :

```
UIManager.put("Button.arc", 20);           // Coins arrondis des boutons
UIManager.put("Component.arc", 15);       // Tous les composants
UIManager.put("TextComponent.arc", 10);   // Champs de texte
UIManager.put("Button.font", new Font("Segoe UI", Font.BOLD, 14));
UIManager.put("defaultFont", new Font("Segoe UI", Font.PLAIN, 13));
```

Place ces lignes **juste après** `FlatLightLaf.setup();`

---

5. Exemple complet avec DesignGridLayout + FlatLaf

```
import javax.swing.*;
import net.java.dev.designgridlayout.DesignGridLayout;
import com.formdev.flatlaf.FlatDarkLaf;

public class ExempleFlatLaf {
    public static void main(String[] args) {
        FlatDarkLaf.setup(); // Thème moderne sombre

        JFrame fen = new JFrame("Connexion Oracle (FlatLaf)");
        fen.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JPanel panneau = new JPanel();
        DesignGridLayout layout = new DesignGridLayout(panneau);

        JTextField champUser = new JTextField("scott", 15);
        JPasswordField champMdp = new JPasswordField("tiger", 15);
        JTextField champUrl = new JTextField("jdbc:oracle:thin:@//localhost:
1521/EE.oracle.docker", 25);

        layout.row().center().add(new JLabel("=== LOGIN ==="));
        layout.row().grid(new JLabel("Utilisateur")).add(champUser);
        layout.row().grid(new JLabel("Mot de passe")).add(champMdp);
        layout.row().grid(new JLabel("URL")).add(champUrl);
        layout.row().center().add(new JButton("Connecter"));

        fen.add(panneau);
        fen.pack();
        fen.setLocationRelativeTo(null);
        fen.setVisible(true);
    }
}
```

🎛️6. FlatLaf IntelliJ Themes (optionnel)

Tu peux utiliser des thèmes IntelliJ `.theme.json` (Dracula, Monokai, etc.) :

1. Télécharge sur https://www.formdev.com/flatlaf/themes/
2. Place le fichier `.theme.json` dans ton projet.
3. Charge-le :

```
import com.formdev.flatlaf.IntelliJTheme;

try {
    IntelliJTheme.setup(Exemple.class.getResourceAsStream("/themes/
Dracula.theme.json"));
```

```
} catch (Exception e) {
    e.printStackTrace();
}
```

🧩7. Customisation avancée via `.properties`

Tu peux créer ton propre fichier `custom.properties` pour ajuster finement les couleurs, polices, etc.

Exemple :

```
@baseColor = #202020
@foreground = #FFFFFF
@accentColor = #0099FF
Button.arc = 12
TextComponent.arc = 8
```

Puis dans le code :

```
import com.formdev.flatlaf.FlatPropertiesLaf;

FlatPropertiesLaf.setup("Custom", Exemple.class.getResourceAsStream("/themes/
custom.properties"));
```

🧩8. Intégration avec ton IDE

- **IntelliJ IDEA / NetBeans / Eclipse** : tu peux changer le L&F à chaud pour voir le rendu.
- **FlatLaf IntelliJ Plugin** : ajoute la palette de couleurs FlatLaf directement dans ton IDE.

🖌️9. Points forts

👈Compatible 100 % avec Swing existant
👈Apparence moderne et uniforme (Windows, Linux, macOS)
👈Ultra léger et sans dépendance externe
👈Thèmes personnalisables (JSON, .properties)
👈Supporte HiDPI et écrans 4K

🧩10. Ressources utiles

- 😛Site officiel : https://www.formdev.com/flatlaf/
- 💚Javadoc : https://www.javadoc.io/doc/com.formdev/flatlaf/latest/index.html
- 💅Thèmes : https://www.formdev.com/flatlaf/themes/

11. Exemple d’application complète avec FlatLaf

```
import javax.swing.*;
import com.formdev.flatlaf.FlatIntelliJLaf;

public class ExempleAppFlatLaf {
    public static void main(String[] args) {
        FlatIntelliJLaf.setup();

        JFrame f = new JFrame("FlatLaf Démo");
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JPanel p = new JPanel();
        p.add(new JLabel("Nom:"));
        p.add(new JTextField(10));
        p.add(new JButton("Valider"));

        f.add(p);
        f.pack();
        f.setLocationRelativeTo(null);
        f.setVisible(true);
    }
}
```

🏹Conclusion

FlatLaf modernise instantanément n’importe quelle appli Swing.
Combine-le avec **DesignGridLayout** pour obtenir une interface professionnelle, fluide et réactive sans code complexe.