syncro soft

Open4Tech Summer School 2021

Java Swing Crash Course

Artur Bozieac
Software Developer, Syncro Soft

XML BEAUTIFIER

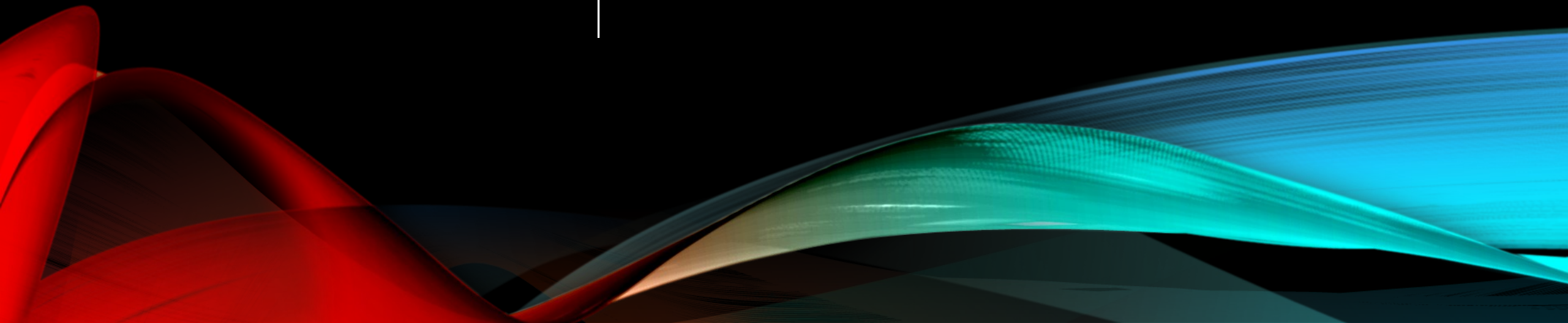
By Bozieac Artur

AGENDA

- Setup pentru Java Development
- Maven si de ce e bine sa il folosim
- XML VS HTML
- GUI cu ajutorul Java Swing
- XML si XSLT si de ce e bine sa le folosim

Setup Guide

SETUP JAVA ENVIRONMENT



Este un instrument de gestionare a proiectelor software.Pe baza conceptului unui model de obiect de proiect (POM), Maven ne ajuta la construirea, raportarea și documentarea unui proiect.

DE CE MAVEN

- Ne permite sa preluam dependente de la alte proiecte
- Ne permite sa aducem dependente in proiectul nostru
- Ne ajuta sa construim proiectul nostru folosind aceeasi configuratie

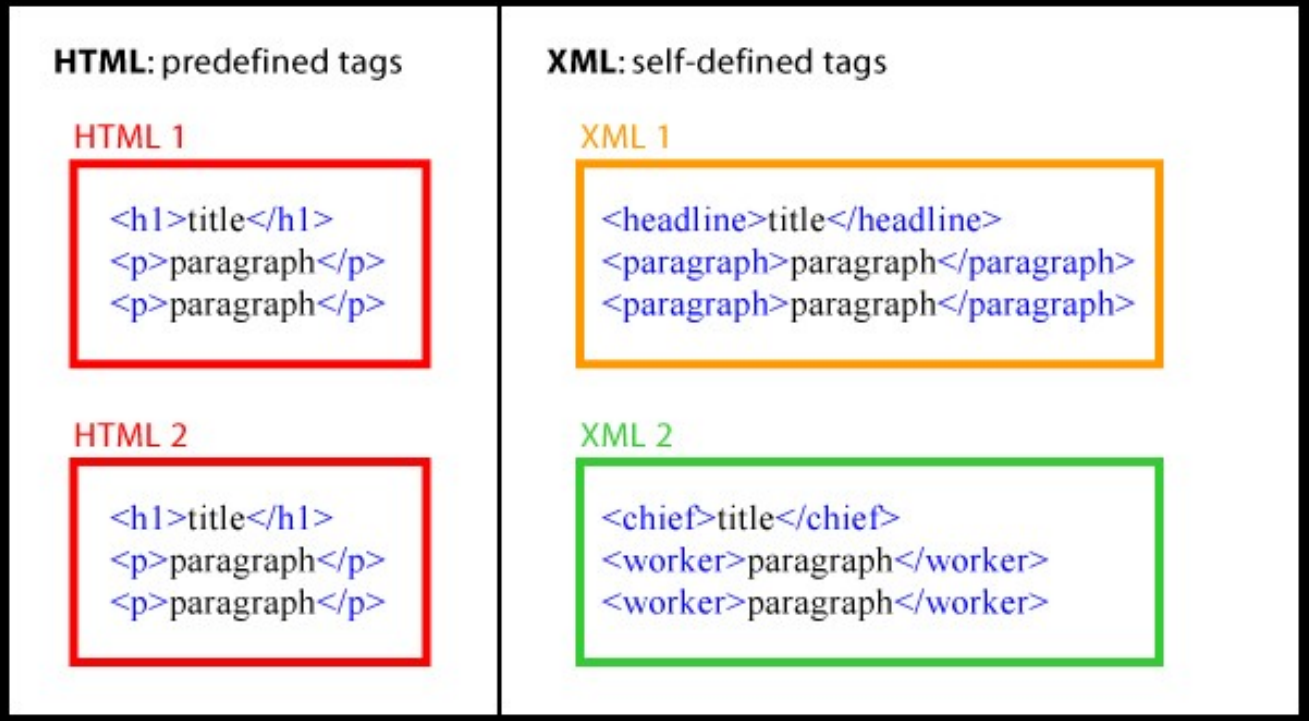
- Guide Create Maven Project
- Maven Repository - Toate dependentele de care ai nevoie
- Guide Maven Achetype

HTML VS XML

- XML - eXtensible Markup Language.
- XML este un tool independent, folosit pentru transmiterea si pastrarea datelor.
- XML ofera un framework pentru definirea limbajului de marcare.
- XML nu are tag-uri predefinite.
- XML este well-formed, necesita respectarea ierarhiei tag-urilor, tag-ul odata deschis, neaparat trebuie sa fie inchis la un moment dat.

- HTML - Hyper Text Markup Language.
- HTML este folosit pentru reprezentarea datelor si este folosit pe cum arata datele.
- HTML este un limbaj de marcare.
- HTML are tag-uri predefinite.
- HTML nu este well-formed, inchiderea tag-urilor nu este obligatorie.

HTML VS XML EXAMPLE



PROJECT OBJECT MODEL(POM)

- JUnit - dependenta pentru teste unitare
- Commons-io – depedenta pentru operatii Input-Output
- Saxon – Saxon is an XSLT and XQuery processor

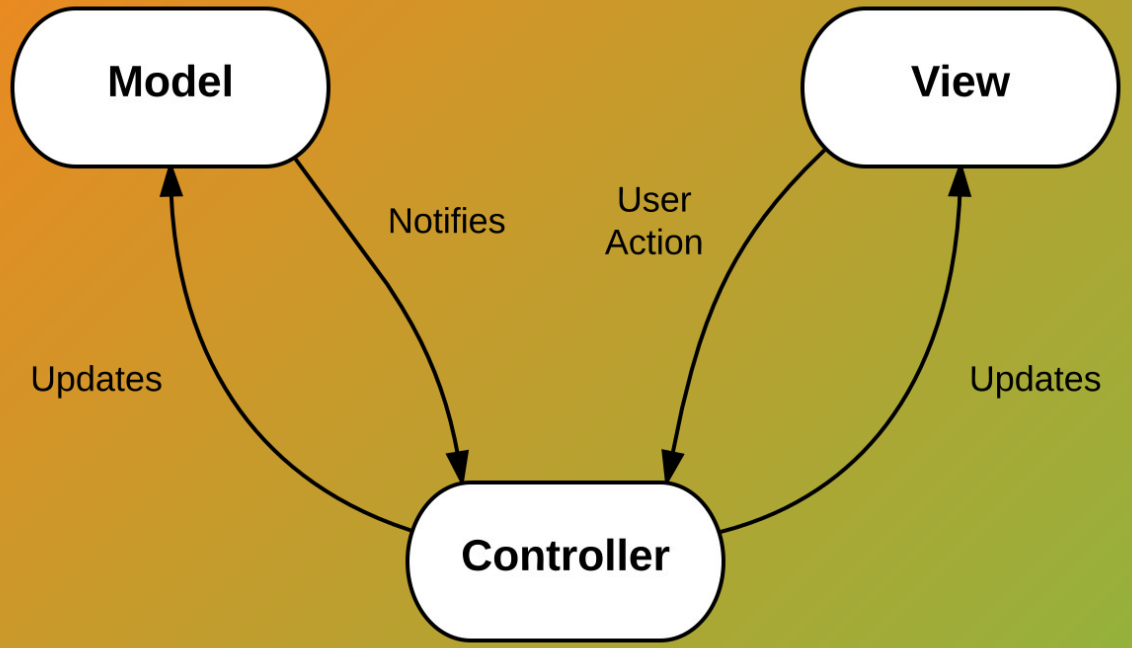
```xml
<project xmlns="http://maven.apache.org/POM/4.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/mave
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.oxygenxml</groupId>
    <artifactId>xml-beautifier</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <dependencies>

        <dependency>
            <groupId>junit</groupId>
            <artifactId>junit</artifactId>
            <version>3.8.1</version>
            <scope>test</scope>
        </dependency>

        <dependency>
            <groupId>commons-io</groupId>
            <artifactId>commons-io</artifactId>
            <version>2.6</version>
        </dependency>

        <!-- https://mvnrepository.com/artifact/net.sf.saxon/Saxon-HE -->
        <dependency>
            <groupId>net.sf.saxon</groupId>
            <artifactId>Saxon-HE</artifactId>
            <version>9.5.1-5</version>
        </dependency>

    </dependencies>
</project>
```

Guide - POM Structure

XML Beautifier

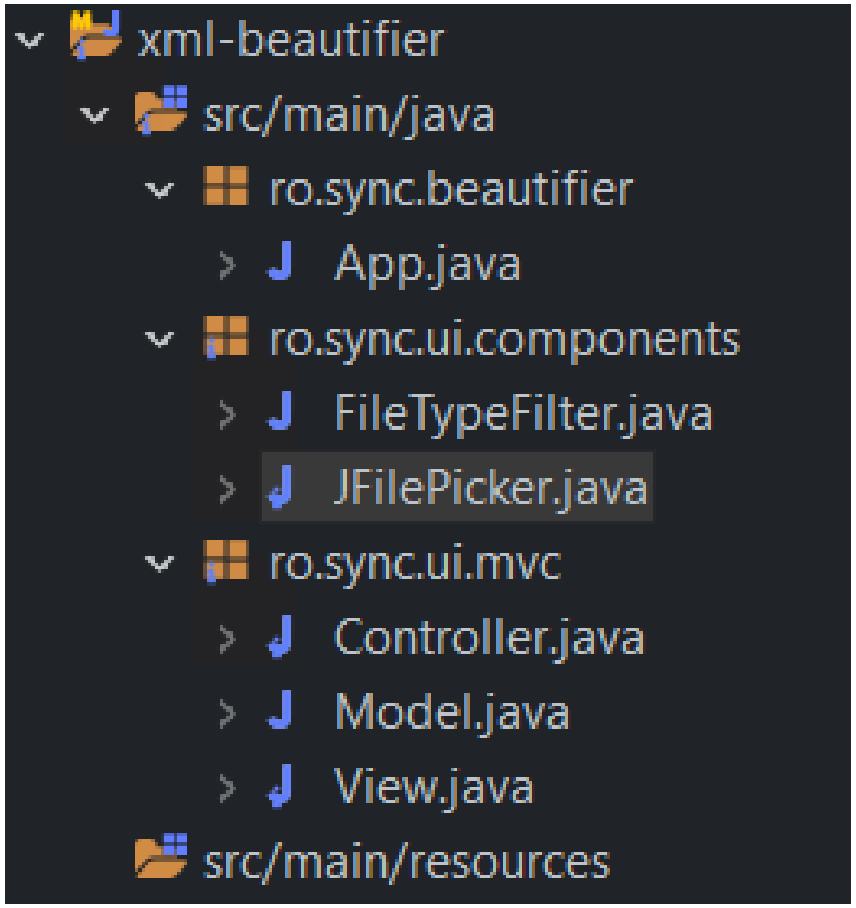
CHECKPOINT

Model View Controller Pattern



MODEL VIEW CONTROLLER WITH JAVA SWING

Guide Clean Code & Design Patterns

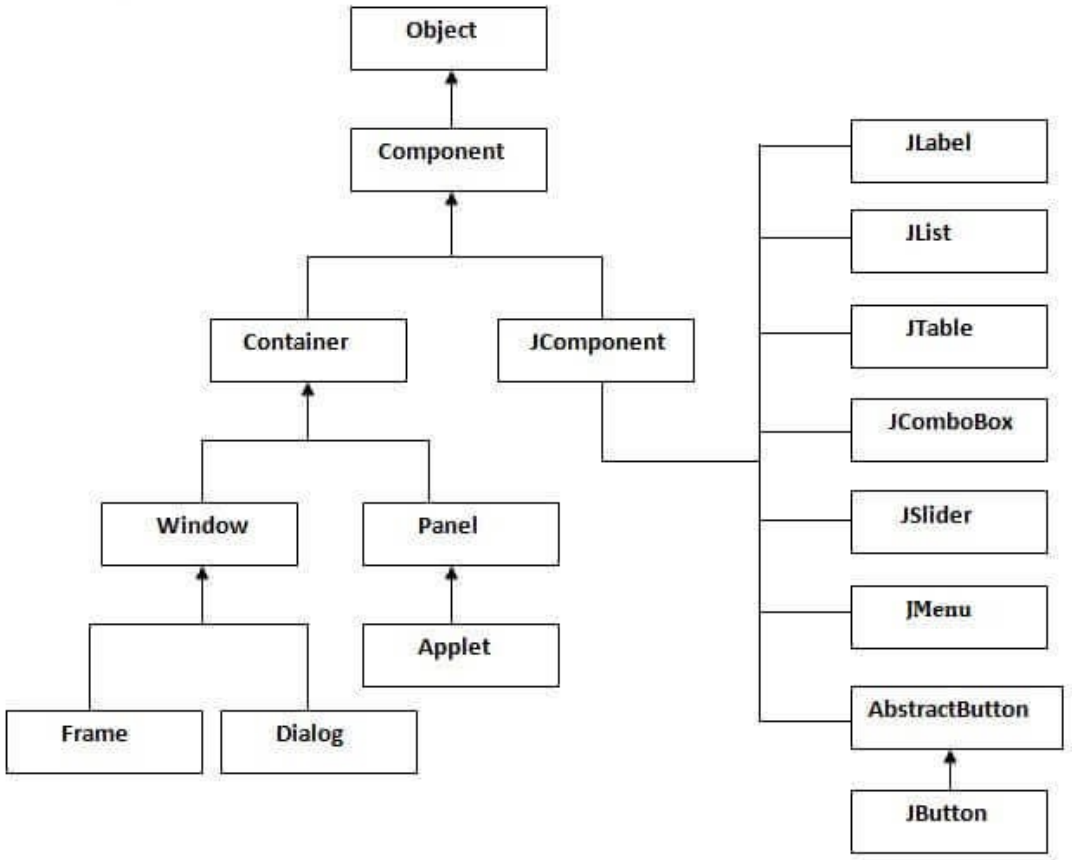


STRUCTURA PROIECTULUI

.

JAVA SWING

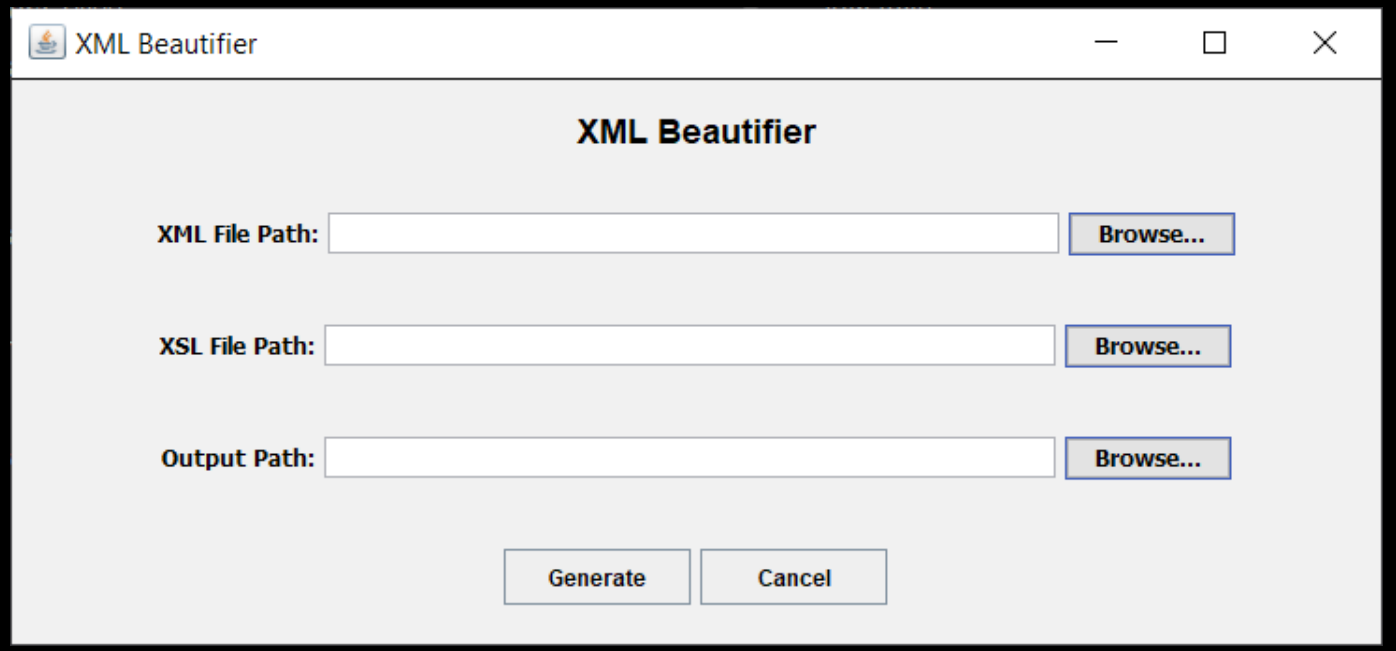
Set de instrumente pentru GUI din Java



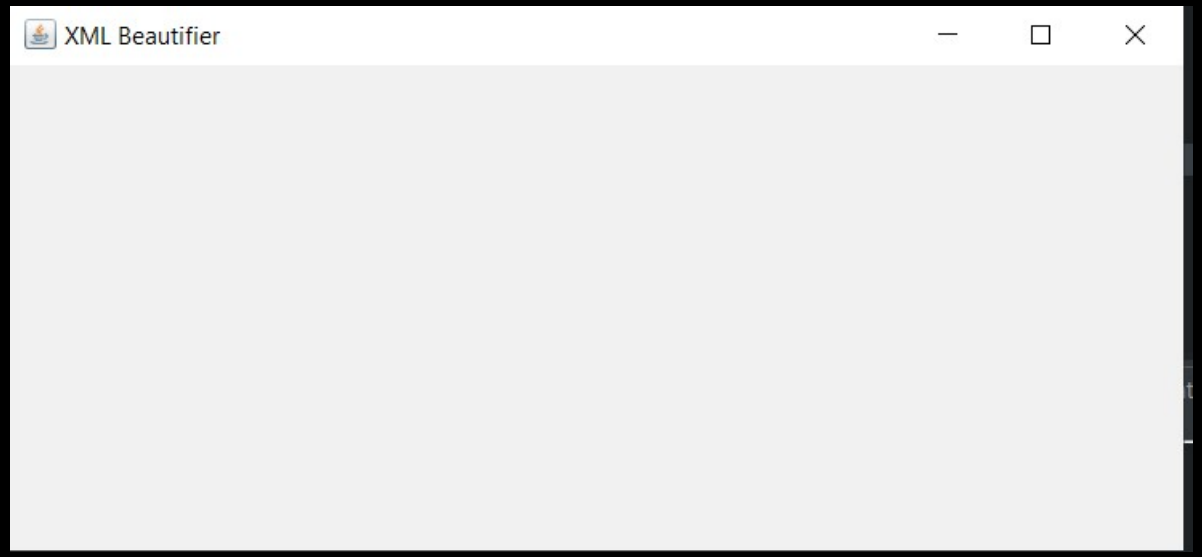
IERARHIA SWING API

COMPONENTE GUI

- Frame
- Butoane
- File Pickere



VIEW - FRAME




```
public class View extends JFrame {

    private static final int WINDOW_WIDTH = 750;
    private static final int WINDOW_HEIGHT = 350;
```

```
/**
 * This method is used to display JFrame Components.
 */
void displayJFrame() {
    // Frame setup stuff

    // Set window title
    this.setTitle(windowTitle);
    // Default action on close operation
    this.setDefaultCloseOperation(WindowConstants.DISPOSE_ON_CLOSE);
    // Window size
    this.setMinimumSize(new Dimension(WINDOW_WIDTH, WINDOW_HEIGHT));
    // Setting relative location to NULL to display window in center
    this.setLocationRelativeTo(null);
    // Window cannot be resized
    this.setResizable(false);
    // MUST Set window visible
    this.setVisible(true);
```

JFILEPICKER

- JFilePicker
  - Jpanel
    - JLabel
    - JTextField
    - JButton

Avem nevoie de o fereastra de deschidere a fisierelor si de una de salvare, definim si un mod de deschidere

```
public class JFilePicker extends JPanel {

    // FilePicker components
    private JLabel label;
    private JTextField textField;
    private JButton button;
    private JFileChooser fileChooser;

    // Unchangeable vars, to switch between Open dialog and Save dialog
    private int mode;
    public static final int MODE_OPEN = 1;
    public static final int MODE_SAVE = 2;
```

JFILEPICKER – FLOWLAYOUT

- Flowlayout – este un layout in care componentele sunt aliniate de la stanga la dreapta
- Constructor : **FlowLayout(int align, int hgap, int vgap)**

```
// setting flowlauout to have elements in one row
FlowLayout flowLayout = new FlowLayout(FlowLayout.CENTER, 5, 5);
flowLayout.setAlignOnBaseline(true);
setLayout(flowLayout);
```

JFILEPICKER

Style

Pozitionare in panel

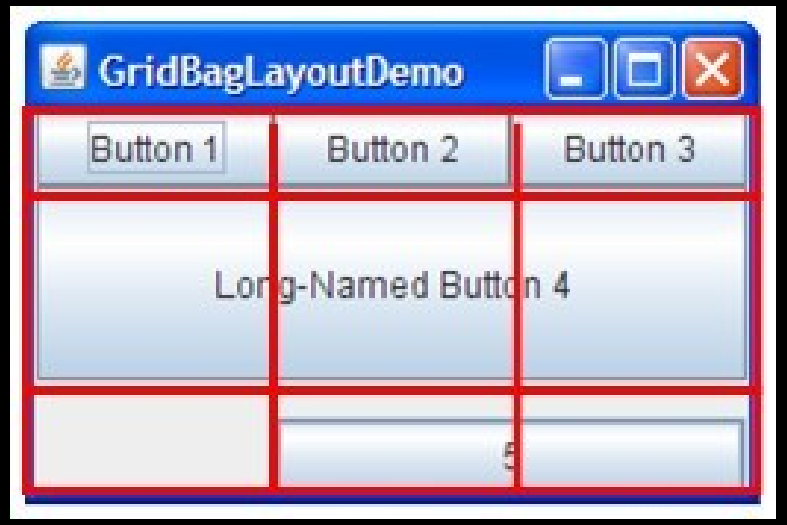
Action listeners

```
    // file chooser instantiation
    fileChooser = new JFileChooser();

    // setting flowlauout to have elements in one row
    FlowLayout flowLayout = new FlowLayout(FlowLayout.CENTER, 5, 5);
    setLayout(flowLayout);

    // creates the GUI ( components styles )
    label = new JLabel(textFieldLabel);
    label.setFont(new Font(Font.SANS_SERIF, Font.BOLD, 12));

    textField = new JTextField(35);

    button = new JButton(buttonLabel);
    button.setForeground(Color.BLACK);
    button.setFont(new Font(Font.SANS_SERIF, Font.BOLD, 12));

    button.addActionListener(new ActionListener() {
        /**
         * Anonymous method which handles action when "browse" button is pushed (
         * buttonActionPerformed()
         *
         * @param evt event which was performed
         */
        public void actionPerformed(ActionEvent evt) {
            buttonActionPerformed(evt);
        }
    });

    // adding elements to layout
    add(label, BorderLayout.WEST);
    add(textField, BorderLayout.CENTER);
    add(button, BorderLayout.EAST);

}
```

CHECKPOINT

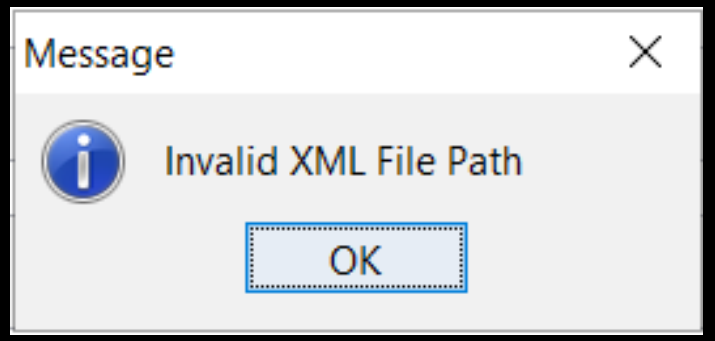
GRIDBAGLAYOUT

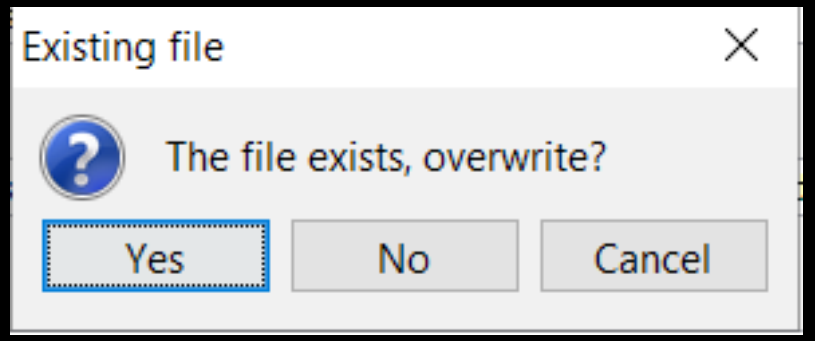
- GridBagLayout este folosit pentru a alinia componentele pe verticală, orizontală sau de-a lungul liniei de bază.



JOPTIONPANE

- Clasa JOptionPane este utilizată pentru a furniza box-uri de dialog standard, cum ar fi box de dialog pentru mesaje, box de confirmare și box de dialog de intrare





XSLT & XPATH

- XPath este un limbaj pentru navigarea în documente XML.
- Exemple :
  - /*/JSONType/properties/person
  - /*/JSONType/version
  - /*/JSONType/usedByValues/usedBy

```
343 <usedByValues>
344 <usedBy href="#/definitions/personType/items/properties/link" name="personType/items/properties/link"></usedBy>
345 <usedBy href="#/definitions/personType/items" name="personType/items"></usedBy>
346 </usedByValues>
```

XSLT & XPATH

- XSLT este un limbaj pentru transformarea documentelor XML.

```
<xsl:template match="/*">
    <html>
        <body>
            <table border="1">
                <thead>
                    <tr>
                        <th>Title</th>
                        <th>Description</th>
                        <th>type</th>
                    </tr>
                </thead>
                <xsl:apply-templates/>
            </table>
        </body>
    </html>
</xsl:template>
```

XML Beautifier

SAXON XSLT

- Saxon este un procesor XSLT

```
<!-- https://mvnrepository.com/artifact/net.sf.saxon/Saxon-HE -->
<dependency>
<groupId>net.sf.saxon</groupId>
<artifactId>Saxon-HE</artifactId>
<version>9.5.1-5</version>
</dependency>
```

TESTARE UNITARA & COMENTARII

- Azi stii perfect ce se intampla in codul tau, te mai intreb o data luna viitoare.

```
button.addActionListener(new ActionListener() {
    /**
     * Anonymous method which handles action when "browse" button is pushed Calls
     * buttonActionPerformed()
     *
     * @param evt event which was performed
     */
    public void actionPerformed(ActionEvent evt) {
        buttonActionPerformed(evt);
    }
});
```

```
/**
 * File Picker constructor (takes 2 strings, one for name of the field, another
 * for button name)
 *
 * @param textFieldLabel represents name of the field
 * @param buttonLabel    represents button name
 */
public JFilePicker(String textFieldLabel, String buttonLabel) {
```

TESTARE UNITARA & COMENTARII

- Testarea unitară este importantă, deoarece developerii încearcă uneori să economisească timp făcând teste unitare minime și acest lucru este mit deoarece testarea unitară necorespunzătoare duce la costuri ridicate. Dacă se realizează testarea corespunzătoare a unității la începutul dezvoltării, atunci se economisește timp și bani în cele din urmă.

INTREBARI ?

Codul sursa: https://github.com/Chandler1215/XML-Beautifier

MULTUMESC PENTRU ATENTIE

Email : artur_bozieac@sync.ro