BUDGET CALCULATOR
SOURCE CODE

```
import java.awt.*;
import java.awt.event.*;
import java.util.ArrayList;

class Transaction {
private String description;
private double amount;
private boolean isIncome;

public Transaction(String description, double amount, boolean isIncome) {
this.description = description;
this.amount = amount;
this.isIncome = isIncome;
}

public String getDescription() {
return description;
}

public void setDescription(String description) {
this.description = description;
}

public double getAmount() {
return amount;
    }
```

```
public void setAmount(double amount) {
this.amount = amount;
}

public boolean isIncome() {
return isIncome;
}

public void setIncome(boolean isIncome) {
this.isIncome = isIncome;
}

@Override
public String toString() {
String type = isIncome ? "Income" : "Expense";
return type + " | " + description + " | $" + amount;
}
}

public class BudgetCalculatorAWT extends Frame {
private ArrayList<Transaction> transactions = new ArrayList<>();
private List transactionList;
private TextArea summaryArea;

public BudgetCalculatorAWT() {
setTitle("Budget Calculator");
setSize(600, 400);
setLayout(new BorderLayout());
```

```
// Transaction List Panel
Panel listPanel = new Panel(new BorderLayout());
transactionList = new List();
listPanel.add(new Label("Transactions:"), BorderLayout.NORTH);
listPanel.add(transactionList, BorderLayout.CENTER);

// Buttons Panel
Panel buttonPanel = new Panel(new GridLayout(1, 5));
Button addButton = new Button("Add");
Button editButton = new Button("Edit");
Button deleteButton = new Button("Delete");
Button viewSummaryButton = new Button("Summary");
Button exitButton = new Button("Exit");

buttonPanel.add(addButton);
buttonPanel.add(editButton);
buttonPanel.add(deleteButton);
buttonPanel.add(viewSummaryButton);
buttonPanel.add(exitButton);

// Summary Panel
summaryArea = new TextArea();
summaryArea.setEditable(false);

// Add Panels to Frame
add(listPanel, BorderLayout.CENTER);
add(buttonPanel, BorderLayout.SOUTH);
add(summaryArea, BorderLayout.EAST);
```

```
// Add Button Listeners
addButton.addActionListener(e -> addTransactionDialog());
editButton.addActionListener(e -> editTransactionDialog());
deleteButton.addActionListener(e -> deleteTransaction());
viewSummaryButton.addActionListener(e -> viewSummary());
exitButton.addActionListener(e -> System.exit(0));

setVisible(true);
}

private void addTransactionDialog() {
Dialog dialog = new Dialog(this, "Add Transaction", true);
dialog.setSize(400, 300);
dialog.setLayout(new GridLayout(5, 2));

TextField descriptionField = new TextField();
TextField amountField = new TextField();
CheckboxGroup typeGroup = new CheckboxGroup();
Checkbox incomeCheckbox = new Checkbox("Income", typeGroup, true);
Checkbox expenseCheckbox = new Checkbox("Expense", typeGroup, false);

dialog.add(new Label("Description:"));
dialog.add(descriptionField);
dialog.add(new Label("Amount:"));
dialog.add(amountField);
dialog.add(new Label("Type:"));
Panel typePanel = new Panel(new GridLayout(1, 2));
typePanel.add(incomeCheckbox);
```

```
typePanel.add(expenseCheckbox);
dialog.add(typePanel);

Button saveButton = new Button("Save");
dialog.add(saveButton);

saveButton.addActionListener(e -> {
try {
String description = descriptionField.getText();
double amount = Double.parseDouble(amountField.getText());
boolean isIncome = incomeCheckbox.getState();

transactions.add(new Transaction(description, amount, isIncome));
transactionList.add(transactions.get(transactions.size() - 1).toString());
dialog.dispose();
} catch (NumberFormatException ex) {
new Dialog(this, "Error", true).add(new Label("Invalid input!"));
}
});

dialog.setVisible(true);
}

private void editTransactionDialog() {
int selectedIndex = transactionList.getSelectedIndex();
if (selectedIndex == -1) {
showErrorDialog("No transaction selected!");
return;
}
```

```
Transaction transaction = transactions.get(selectedIndex);

Dialog dialog = new Dialog(this, "Edit Transaction", true);
dialog.setSize(400, 300);
dialog.setLayout(new GridLayout(5, 2));

TextField descriptionField = new TextField(transaction.getDescription());
TextField amountField = new
TextField(String.valueOf(transaction.getAmount()));
CheckboxGroup typeGroup = new CheckboxGroup();
Checkbox incomeCheckbox = new Checkbox("Income", typeGroup,
transaction.isIncome());
Checkbox expenseCheckbox = new Checkbox("Expense", typeGroup,
!transaction.isIncome());

dialog.add(new Label("Description:"));
dialog.add(descriptionField);
dialog.add(new Label("Amount:"));
dialog.add(amountField);
dialog.add(new Label("Type:"));
Panel typePanel = new Panel(new GridLayout(1, 2));
typePanel.add(incomeCheckbox);
typePanel.add(expenseCheckbox);
dialog.add(typePanel);

Button saveButton = new Button("Save");
dialog.add(saveButton);
```

```
saveButton.addActionListener(e -> {
try {
transaction.setDescription(descriptionField.getText());
transaction.setAmount(Double.parseDouble(amountField.getText()));
transaction.setIncome(incomeCheckbox.getState());

transactionList.replaceItem(transaction.toString(), selectedIndex);
dialog.dispose();
} catch (NumberFormatException ex) {
showErrorDialog("Invalid input!");
}
});

dialog.setVisible(true);
}

private void deleteTransaction() {
int selectedIndex = transactionList.getSelectedIndex();
if (selectedIndex == -1) {
showErrorDialog("No transaction selected!");
return;
}

transactions.remove(selectedIndex);
transactionList.remove(selectedIndex);
}

private void viewSummary() {
double totalIncome = 0;
```

```
double totalExpenses = 0;

for (Transaction transaction : transactions) {
if (transaction.isIncome()) {
totalIncome += transaction.getAmount();
} else {
totalExpenses += transaction.getAmount();
}
}

double balance = totalIncome - totalExpenses;

summaryArea.setText("Total Income: $" + totalIncome + "\n" +
"Total Expenses: $" + totalExpenses + "\n" +
"Balance: $" + balance);
}

private void showErrorDialog(String message) {
Dialog dialog = new Dialog(this, "Error", true);
dialog.setSize(300, 100);
dialog.setLayout(new FlowLayout());
dialog.add(new Label(message));
Button closeButton = new Button("Close");
closeButton.addActionListener(e -> dialog.dispose());
dialog.add(closeButton);
dialog.setVisible(true);
}
```