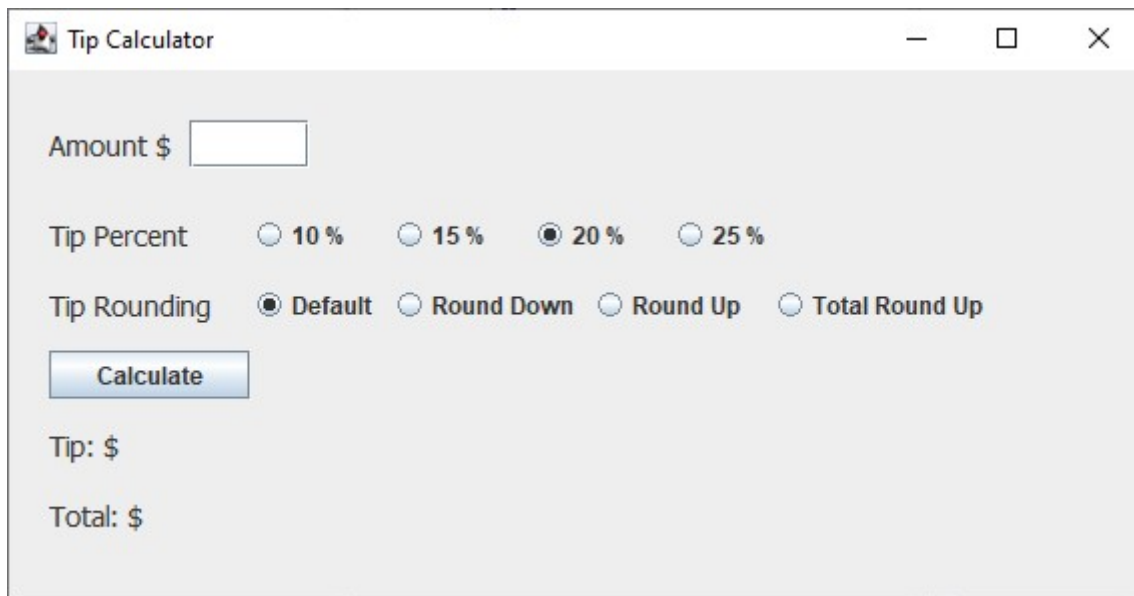


# Lab3: Tip Calculator

---




The screenshot shows a Java Swing window titled "Tip Calculator". It contains a text field for "Amount \$", a group of radio buttons for "Tip Percent" (10%, 15%, 20%, 25%), and another group of radio buttons for "Tip Rounding" (Default, Round Down, Round Up, Total Round Up). A "Calculate" button is positioned below the rounding options. At the bottom, there are labels for "Tip: \$" and "Total: \$".

## Objectives

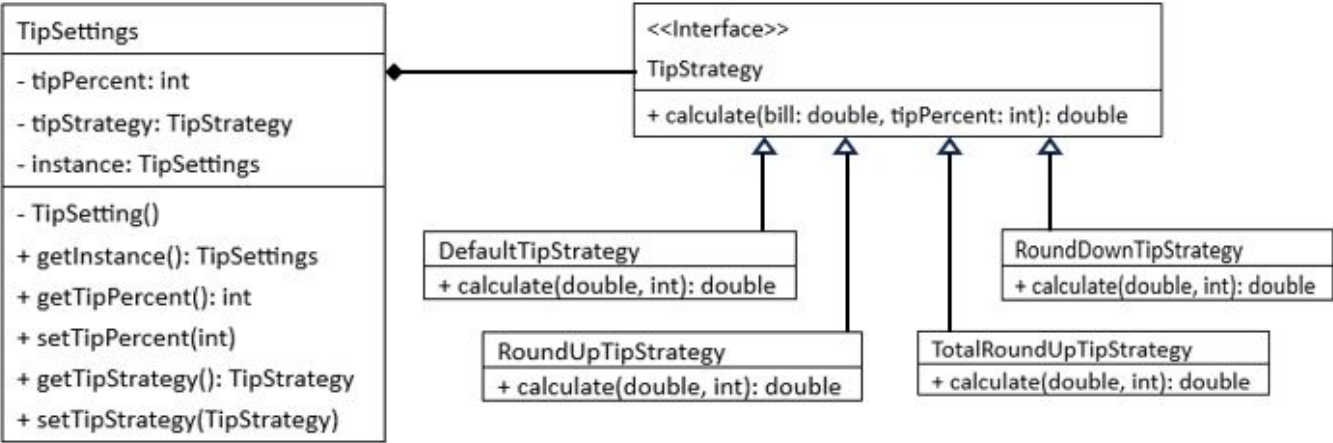
- Learn how to use and implement Singleton Design Pattern
- Learn how to use and implement Strategy Design Pattern

## Work

1. Tip Calculator calculates the tip based on the selection of tip percent and tip rounding for the amount of bill.
2. The program uses the singleton design pattern and the strategy design pattern.
3. `Lab3.java` is the main Java file which defines all the window components. You don't need to make any changes to the file.
4. `TipSetting.java` is a singleton class file.
5.  **Complete the `TipSettings` class in the class file. Its class diagram is shown below.**

TipSettings
- tipPercent: int - tipStrategy: TipStrategy - instance: TipSettings
- TipSetting() + getInstance(): TipSettings + getTipPercent(): int + setTipPercent(int) + getTipStrategy(): TipStrategy + setTipStrategy(TipStrategy)

4. `TipStrategy` is defined as an interface in `TipStrategy.java`. It is implemented in four different tip strategy classes: `TipStrategyDefault`, `TipStrategyRoundDown`, `TipStrategyRoundUp`, and `TipStrategyTotalRoundUp`. The first three tip strategies are defined in their class files.



5. The tip calculation of `TipStrategyTotalRoundUp` is as follows:

- `tip = bill * tipPercent / 100`
- `total = Ceiling(tip + bill) // use Math.ceil`
- `finalTip = total - bill`

6.  **Complete the `TipStrategyTotalRoundUp` class in `TipStrategyTotalRoundUp.java`.**

---

End of Lab3