Southern New Hampshire University

IT 511 Stepping Stone Overview

Your work in this course includes a series of stepping stone labs that provide important practice in building key skills of object-oriented programming that you will use in the final project. This table provides an overview of the stepping stone sequence, including a description of each lab and its location in the course. A description of how to connect your work in each stepping stone to the final project is also provided.

The two final project milestones are also described here so that you can see how they also fit into your work for the course and help you prepare for the final project.

The UML code for the course and stepping stone code completion and submission procedures are also included in this document.

Task	Module	Stepping Stone Description	Final Project Connection
Stepping Stone Lab One: Pseudocode	**Two**	In Stepping Stone Lab One, you will write pseudocode for your collection manager program.	The pseudocode is for a program that is similar to the one in the final project. In this case/scenario, the student is helping a friend organize a collection.
Stepping Stone Lab Two: Data Types	**Three**	In Stepping Stone Lab Two, you will begin working with a recipe manager program by focusing on a single ingredient. You will be introduced to basic data types to store numeric values and string values. You will build this ingredient in code that reflects variables such as its name, the number of cups of that ingredient needed in the recipe, and how many calories it has per cup. You will also write an expression to calculate the total number of calories this ingredient would contribute to a recipe. To keep track of this information, you need to store it in a program. You will need to utilize variables of various data types to store the information and prompt the user for the various values. We will assign variables of the appropriate data type and values as shown below:	For your final project, you will do the following: 1. Create a new java class named Ingredient 2. Adapt the code from this Steppingstone to include: A. Change numberCups to represent a more general measurement. B. Add a variable to record the type of measurement (cups, ounces, etc.). C. Prompt the user to input the type of measurement.

<table>
<tr>
<td></td>
<td></td>
<td>
<table>
<tr><th>Variable Name</th><th>Type of Value Stored</th></tr>
<tr><td>Name of ingredient</td><td>Words</td></tr>
<tr><td>Number of cups</td><td>Decimal numbers (e.g., ½ a cup)</td></tr>
<tr><td>Number of calories per cup</td><td>Whole numbers</td></tr>
<tr><td>Total calories</td><td>Decimal numbers</td></tr>
</table>
<p>Once we get that information from the user, we can calculate the total number of calories per serving in the recipe.</p>
</td>
<td></td>
</tr>
<tr>
<td><b>Stepping Stone Lab Three: Branches</b></td>
<td><b>Four</b></td>
<td>
<p>In this module, you have studied how to use conditionals and logical structures to create branches or “forks in the code.” In Stepping Stone Lab Three, you will put this emerging knowledge into practice by developing a branching structure for the recipe manager program.</p>
<p>Be sure to review the Stepping Stone Three guidelines before beginning this lab. Remember, this document also includes completed code from Stepping Stone Lab Two. This code serves a number of functions:</p>
<ul>
<li>It allows you to review your own submitted code.</li>
<li>You may use it as a useful foundation for Stepping Stone Three as well as other assignments to come.</li>
<li>Together with instructor feedback on your submitted code, you are building your own final project application.</li>
</ul>
<p>In Stepping Stone Lab Three, you will write a short application that uses conditionals to create a “forked” branching structure for the recipe manager.</p>
</td>
<td>
<p>For your final project:</p>
<p>Adapt your Ingredient Java file to include data-type validation steps for each of the variables in the class:</p>
<ul>
<li>ingredientName</li>
<li>ingredientAmount</li>
<li>unitMeasurement</li>
<li>ingredientCalories</li>
</ul>
</td>
</tr>
</table>

		Whenever a program accepts user input, it is best practice to be sure the input is what you as the programmer expect. As you continue to develop the recipe manager, you will need to be able to validate user input and ensure the user enters values that are valid. In this lab, you will write a short program that first tests that the input is numerical, then checks that the number is within a specific range, and checks that the maximum number of cups of our main ingredient is 100 and that the minimum number of cups is 1. This application uses the Scanner class to accept a number between 1 and 100 from the user. The Scanner class is useful for parsing primitive values, including numbers. Specifically, you will create a branching structure that leads to the following output: If the number entered is between 1 and 100 (inclusive), the application will display a message that says, “Good job! The number you entered is___.” However, if the number entered is not between 1 and 100 (inclusive), an error message will be displayed to inform the user that the entry does not fit the expected range: “The number entered was not between 1 and 100!”	
Stepping Stone Lab Four: Loops	**Five**	In Module Five, you have studied how to code for iteration through the use of loops. In Stepping Stone Lab Four, you will develop a simple program with a loop structure. Then, you will reflect on how loops may be used to help structure the program you have been working with throughout the stepping stones, designed to organize and manage your friend’s collection. Be sure to review the Stepping Stone Lab Four guidelines before beginning this lab. As well as providing assignment information and the grading rubric, this document includes corrected code	Milestone One: Final Project Ingredient Class

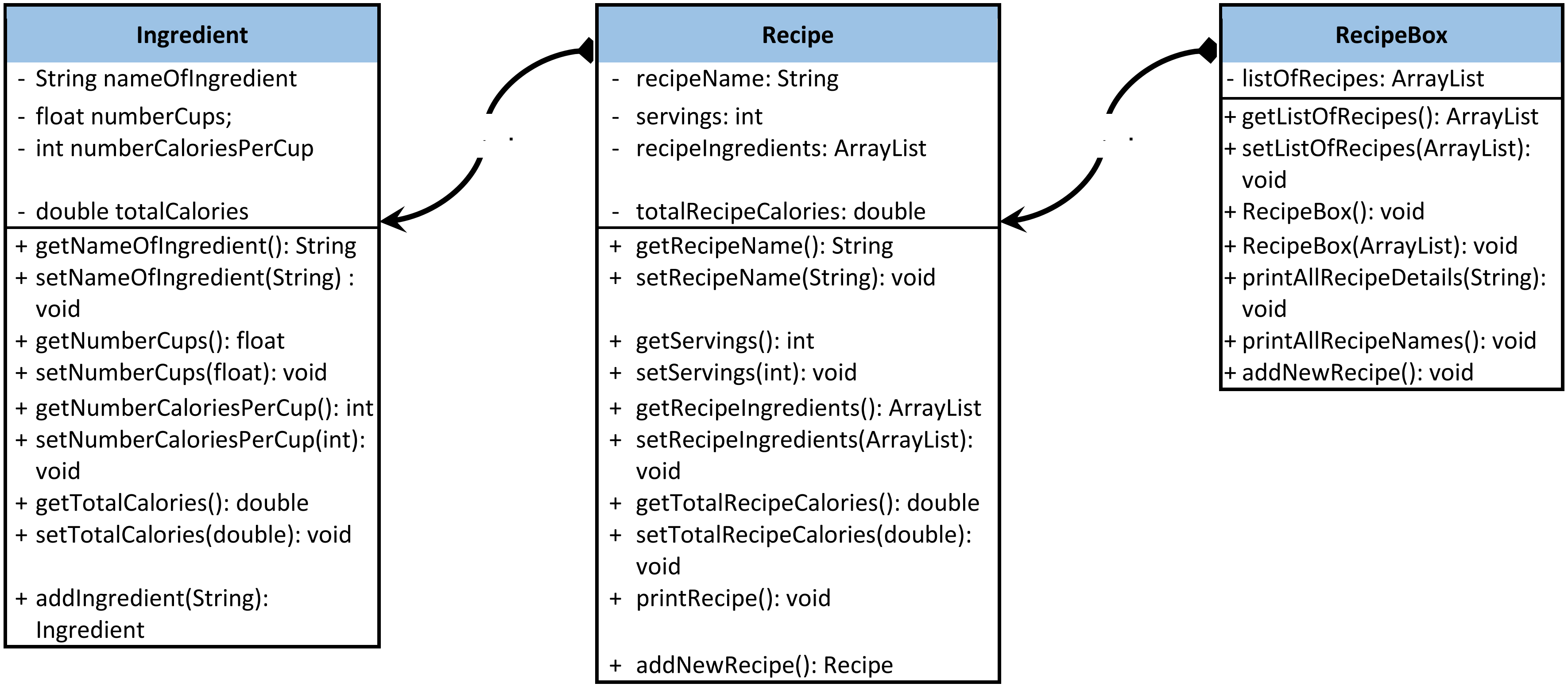
		from Stepping Stone Lab Three, which may serve as a useful foundation for Stepping Stone Lab Four as well as for other assignments to come. The code for this stepping stone lab is short, but the concepts are extremely important. You will use two different looping techniques: a “do” loop and a “for” loop. This stepping stone lab also includes your first application of the Java ArrayList data type. In this stepping stone lab, you will use the ArrayList to store strings, but in future submissions, you will replace these with object types (ingredients). In this lab, you will write a short Java application that uses a loop control structure. • When, or in the terms of object-oriented programming, “while,” the input entered is **not** a specific value (“end”), the program will loop back to prompt the user for another input—either a valid value or an additional ingredient. • If the value entered is “n,” the application will display a list of the ingredients entered. • Do not use a break statement in your program. In your reflection, discuss how loops may be developed to add utility to a program developed to manage your friend’s collection. Describe the added functionality with specific examples.	
Final Project Milestone One: Ingredient Class	**Five**	Your Ingredient class will model the details of individual ingredients in a recipe. Based on Stepping Stone Labs Two and Three, you will create an Ingredient class and give it the basic attributes: name, amount, unit of measure, and calories. Additionally, you will add code to validate the data type of the user input. This Ingredient class will be modified for the submission of your final RecipeManager application; however, it should be functional code that accepts user input for each variable.	In your final project, you are creating a program that will help you manage a collection of items. To complete this program, you will implement two classes: one for the main item and one for the entire collection. If you decide to be more adventurous, you can make an additional class for

			the most important subcomponent of your main item class.
Stepping Stone Lab Five: Recipe Class With Accessors and Mutators	**Six**	Now that you have some experience building basic, single-class applications, here is an opportunity to delve into making a full-fledged application with more than one class. You will start to implement a first version of the Recipe class and create a test class to put it through its paces. You will create the instance variables you need (the recipeName and the mainIngredient details) as well as the methods (the accessors/mutators, the constructors, and the extra print details method) for the Recipe class. Then, you will create a driver, or test class, that will create a Recipe object and try out a few of its methods. This stepping stone lab is the point where we start to pull all the elements of the course together. You will build a Recipe class. In the stepping stone lab assignment, you will get user input to collect the recipe name and serving size, using the ingredient entry code from SS4 to add ingredients to the recipe and calculating the calories per serving. Additionally, you will build your first custom method to print the recipe to the screen. As you are developing the code in this stepping stone lab, you should consider how you can transition it to your final project. Eventually, the user input for collecting the ingredients into an ArrayList of strings will be converted to an ArrayList of Ingredient objects. Think about the variable names you are using and where in the code you are collecting the ingredient input.	For your final project: • Change the ArrayList type from String to an Ingredient object. When a user adds an ingredient to the recipe, instead of adding just the ingredient name, you will add the actual ingredient including name, amount, and calories. • Adapt the printRecipe() method to print the amount and unit of measurement as well as the ingredient name. • Create a custom method in the Recipe class. Choose one of the following options: A. Print out a recipe with amounts adjusted for a different number of servings. B. Create an additional list or ArrayList that allows users to insert step-by-step recipe instructions. C. Convert the ingredient amounts from English to metric (or vice versa). D. Calculate select nutritional information. E. Calculate recipe cost. F. Propose a suitable alternative to your instructor.

Final Project Milestone Two: Recipe Class	**Seven**	In your final project, you are creating a program that will help you manage a collection of recipes. The Recipe class you will build for this milestone will hold all the details of the recipe, the methods to create a new recipe, and a method to print a recipe. In your final project submission, this class will also contain a custom method to add a new feature. In your submission for Milestone Two, you will include commented out pseudocode for this method. In this milestone, you submit the final project version of your Recipe class. Your submission should include the Recipe.java file and a Recipe_Test.java file. Your Recipe class should include the following items: • Instance variables : recipeName, servings, recipeIngredients, and totalRecipeCalories • Accessors and mutators for the instance variables • Constructors • A printRecipe() method • An createNewRecipe() method to build a recipe from user input • Pseudocode for the custom method selected from the list in Stepping Stone Lab Five Your Recipe_Test.java file with a main() method that does the following: • Uses a constructor to create a new recipe • Accesses the printRecipe() method to print the formatted recipe • Invokes the createNewRecipe()method to accept user input	
Stepping Stone Lab Six: RecipeBox	**Eight**	In this stepping stone lab, you will create the RecipeBox driver application. This class will be the class that contains the only “public static void main(String[] args)” method and this class will	For your final project submission: Add a menu item and a method to access the custom method you chose

Driver Application		hold any Recipe objects you add to it. Your submission should create the custom methods printAllRecipeDetails(), printAllRecipes(), and addRecipe().	and developed for the Recipe class based on the Stepping Stone Five Lab.
Final Project Submission	**Nine**	You will need to add Javadoc documentation to the program and ensure the program is a fully functional program, including a test class that shows a sample run of the class that demonstrates all the public methods of your final classes.	

UML Overview



Ingredient

- String nameOfIngredient
- float numberCups;
- int numberCaloriesPerCup
- double totalCalories

+ getNameOfIngredient(): String
+ setNameOfIngredient(String) : void
+ getNumberCups(): float
+ setNumberCups(float): void
+ getNumberCaloriesPerCup(): int
+ setNumberCaloriesPerCup(int): void
+ getTotalCalories(): double
+ setTotalCalories(double): void
+ addIngredient(String): Ingredient

Recipe

- recipeName: String
- servings: int
- recipeIngredients: ArrayList
- totalRecipeCalories: double

+ getRecipeName(): String
+ setRecipeName(String): void
+ getServings(): int
+ setServings(int): void
+ getRecipeIngredients(): ArrayList
+ setRecipeIngredients(ArrayList): void
+ getTotalRecipeCalories(): double
+ setTotalRecipeCalories(double): void
+ printRecipe(): void
+ addNewRecipe(): Recipe

RecipeBox

- listOfRecipes: ArrayList

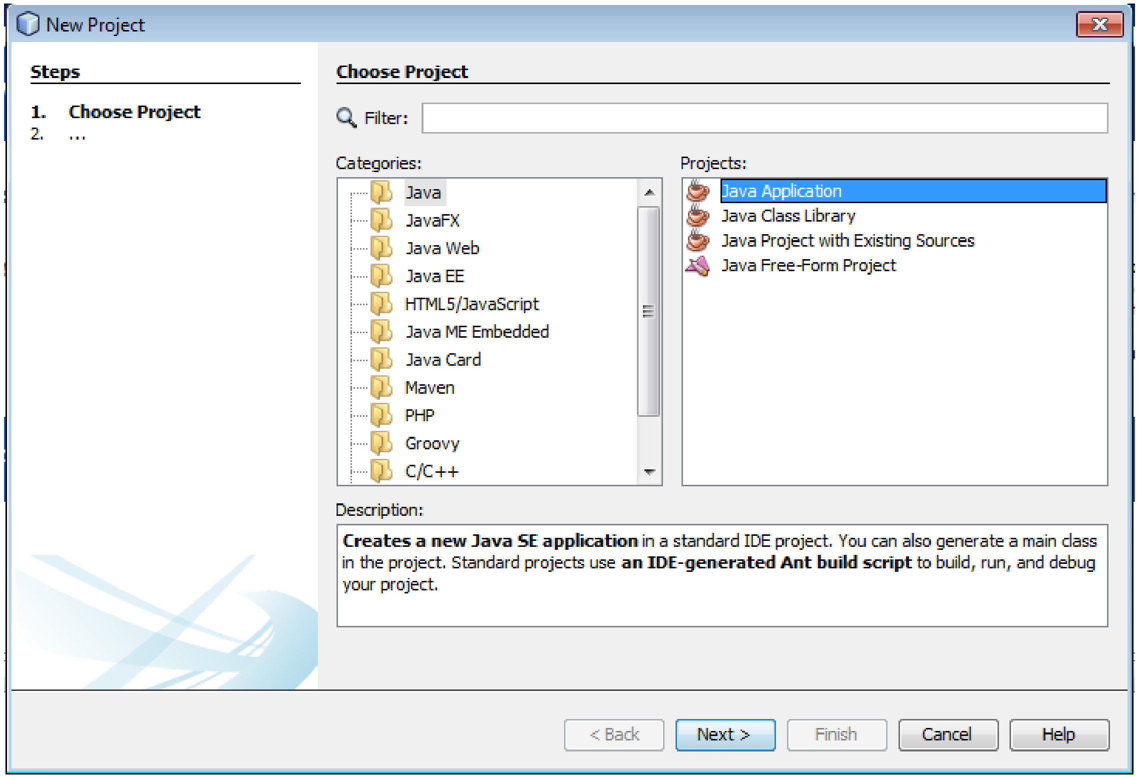
+ getListOfRecipes(): ArrayList
+ setListOfRecipes(ArrayList): void
+ RecipeBox(): void
+ RecipeBox(ArrayList): void
+ printAllRecipeDetails(String): void
+ printAllRecipeNames(): void
+ addNewRecipe(): void

Southern New Hampshire University

Stepping Stone Completion and Submission Directions

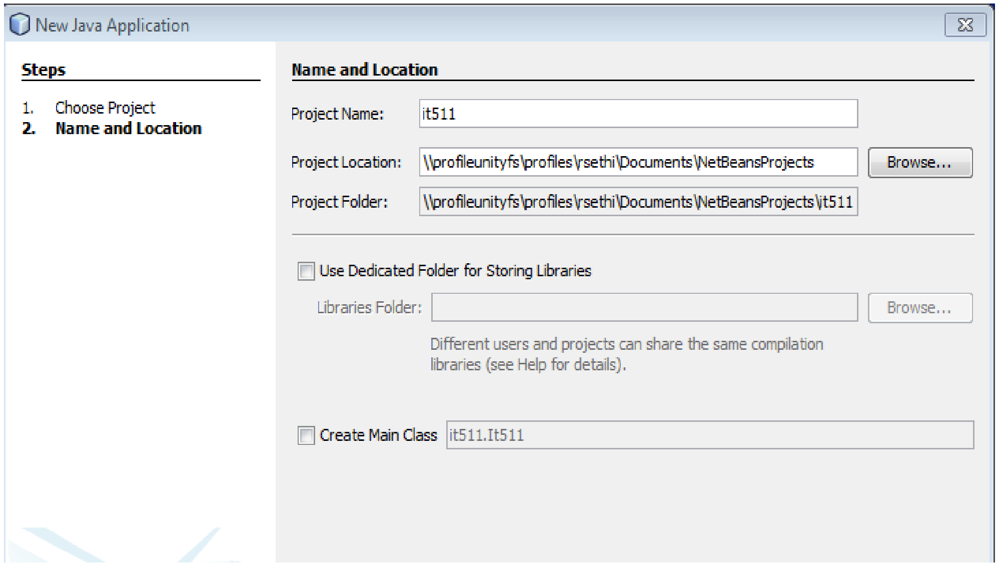
The following are step-by-step directions for completing the stepping stone labs:

1. Launch NetBeans.
2. Choose **File** → **New Project**.



Creates a new Java SE application in a standard IDE project. You can also generate a main class in the project. Standard projects use **an IDE-generated Ant build script** to build, run, and debug your project.

3. Choose a name (it511) and location (leave as default) for it. Also uncheck the “Create Main Class” option.



New Java Application

Steps

1. Choose Project
2. **Name and Location**

Name and Location

Project Name: it511

Project Location: \\profileunityfs\profiles\rsethi\Documents\NetBeansProjects

Project Folder: \\profileunityfs\profiles\rsethi\Documents\NetBeansProjects\it511

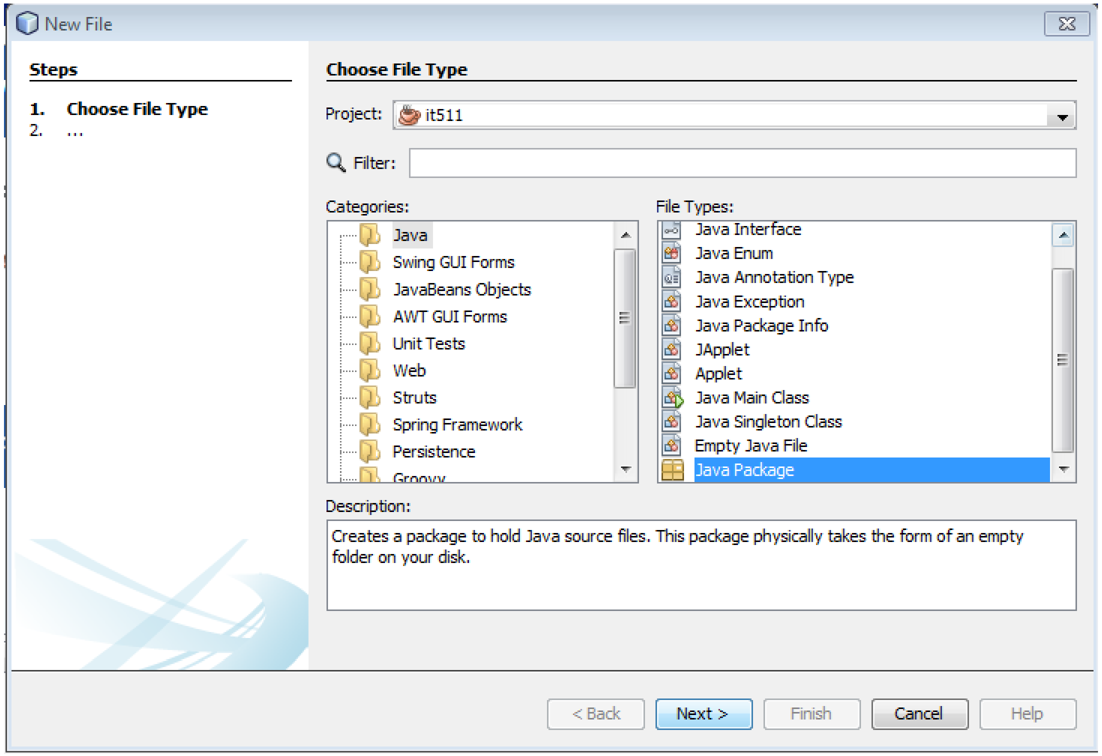
□ Use Dedicated Folder for Storing Libraries

Libraries Folder:

Different users and projects can share the same compilation libraries (see Help for details).

□ Create Main Class it511.It511

4. Next, click your it511 project in the Project Explorer. Then choose **File** → **New File** and scroll down to **Java Package**.



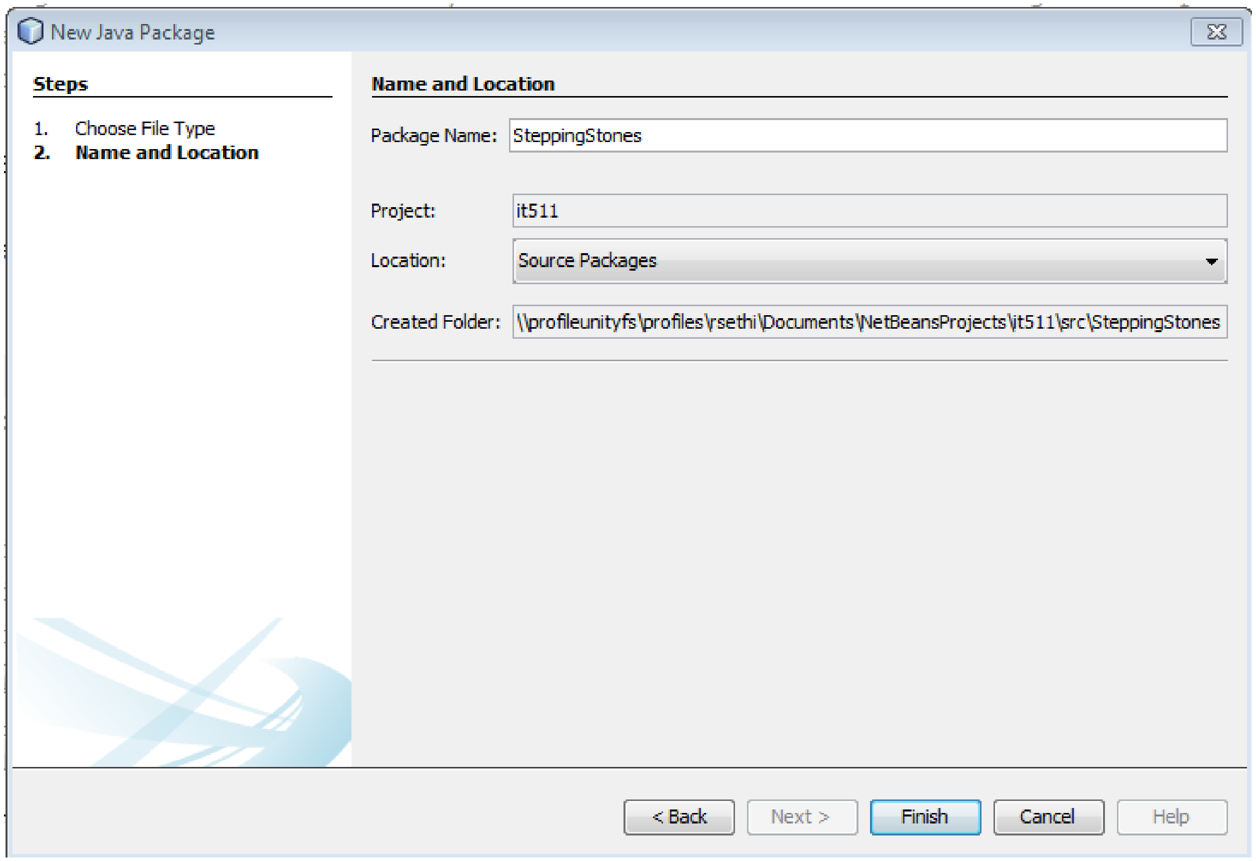
Project: it511

Filter:

Description:

Creates a package to hold Java source files. This package physically takes the form of an empty folder on your disk.

5. On the next screen, set the name of the package. In this case, set it to SteppingStones.



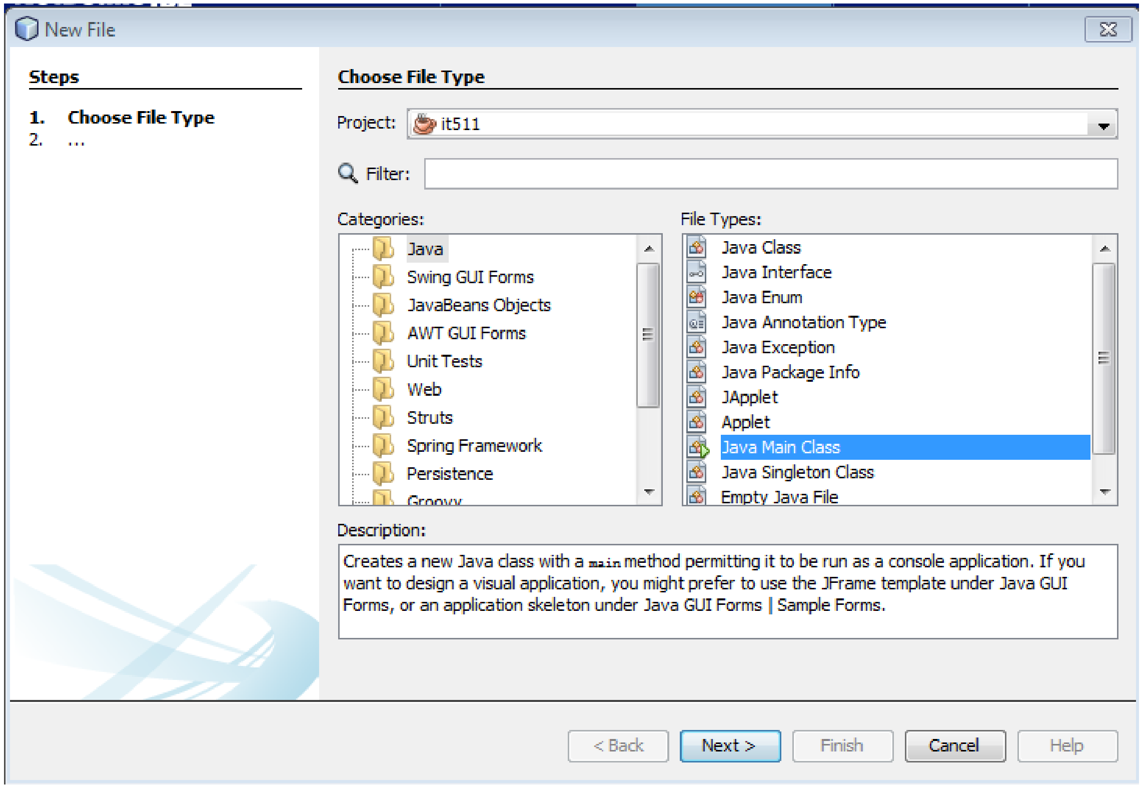
Package Name: SteppingStones

Project: it511

Location: Source Packages

Created Folder: \\profileunityfs\profiles\rsethi\Documents\NetBeansProjects\it511\src\SteppingStones

6. Next, click your new package (SteppingStones). Then choose **File** → **New File** and scroll down to **Java Main Class**.

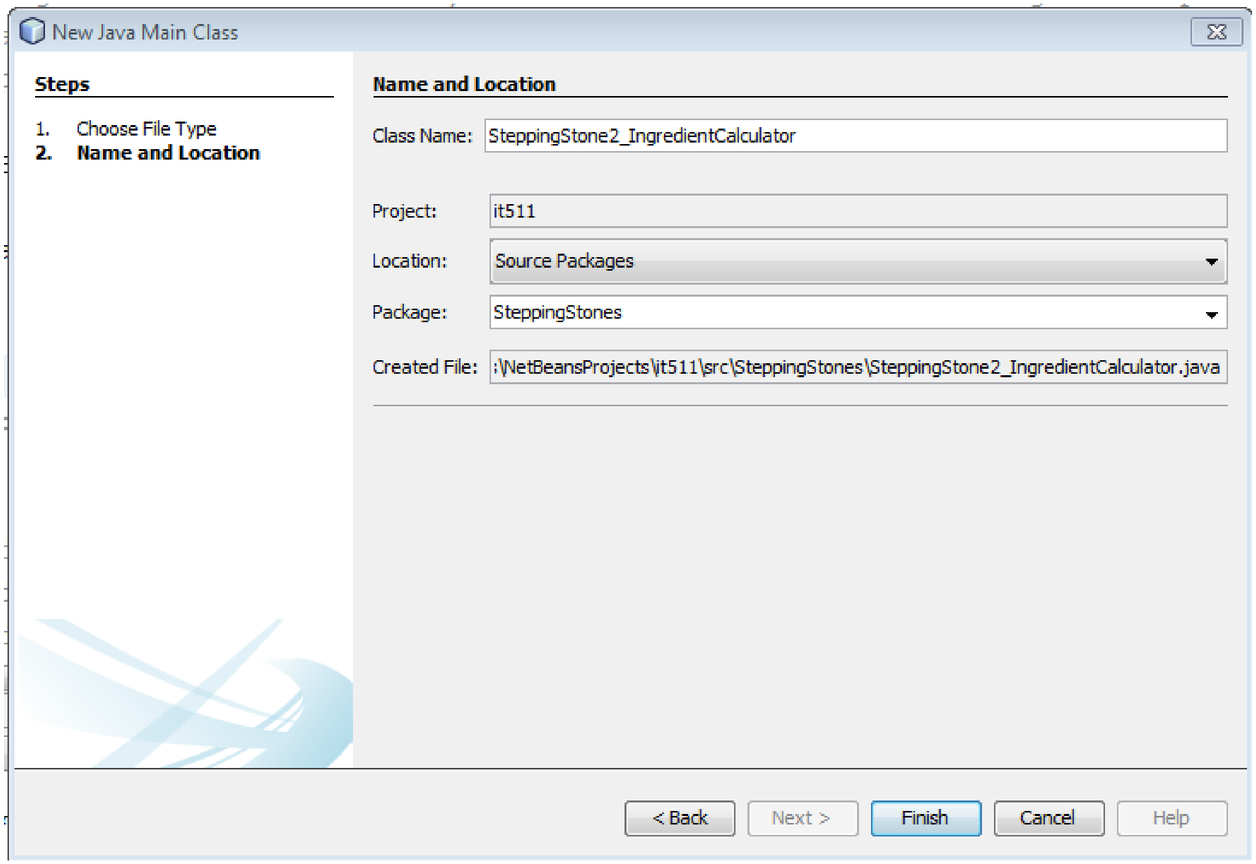


Project: it511

Description:

Creates a new Java class with a main method permitting it to be run as a console application. If you want to design a visual application, you might prefer to use the JFrame template under Java GUI Forms, or an application skeleton under Java GUI Forms | Sample Forms.

7. On the next screen, give the new main class a name (in the case of Stepping Stone Two: SteppingStone2_IngredientCalculator).



Class Name: SteppingStone2_IngredientCalculator

Project: it511

Location: Source Packages

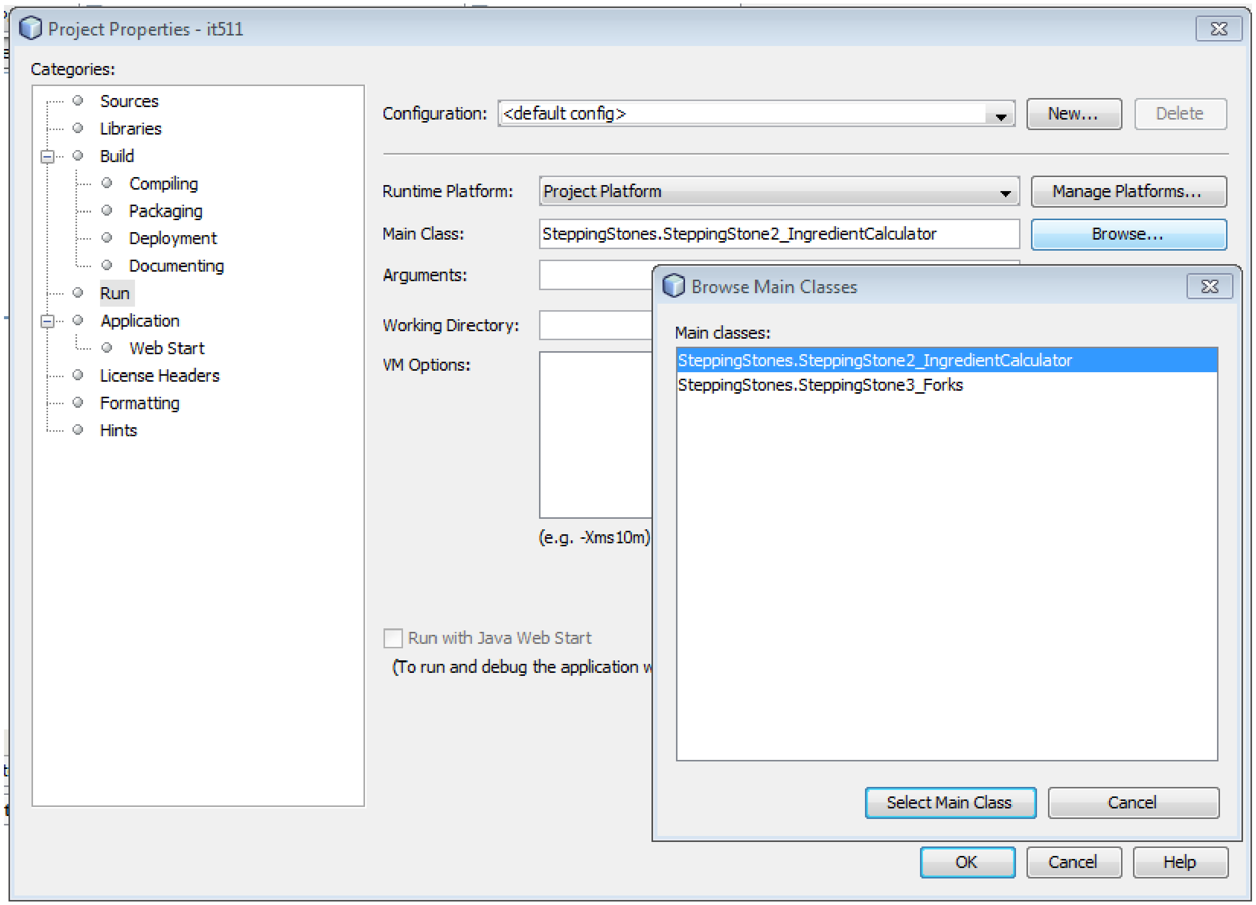
Package: SteppingStones

Created File: :\NetBeansProjects\it511\src\SteppingStones\SteppingStone2_IngredientCalculator.java

8. Now you are ready to enter your code and run it.

9. When you are done with the first stepping stone, repeat steps 6 through 8 above to create your next stepping stone.

10. Once you have created your next stepping stone (e.g., SteppingStone3_Forks), you will need to tell NetBeans which main application class to run when you hit the play button. To do this, right-click your project and choose **Properties**; choose the **Run** menu item, then **Main Class**.



Configuration: <default config>

Runtime Platform: Project Platform

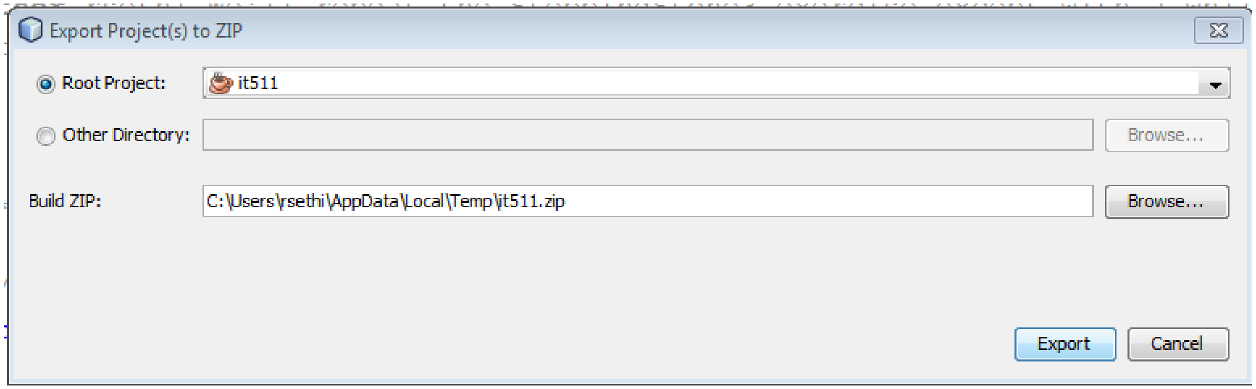
Main Class: SteppingStones.SteppingStone2_IngredientCalculator

Main classes:

SteppingStones.SteppingStone2_IngredientCalculator
SteppingStones.SteppingStone3_Forks

11. In this dialog box, choose the main application class to run when the play button is pressed and click on **Select Main Class**.

12. Finally, when you are ready to submit your project, choose **File** → **Export Project** → **To ZIP...** and you can export the project as a zip file.



Export Project(s) to ZIP

● Root Project: it511

○ Other Directory:

Build ZIP: C:\Users\rsethi\AppData\Local\Temp\it511.zip

13. You are done!