CS 2334 Lab 8 Exceptions

- Exceptions provide a means of handling error conditions that arise in the course of program execution
- Example: The program prompts the user for a filename, and the use provides a nonexistent file
 - Should the program crash?
 - Or would it be better to notify the user and ask again?

Handling Exceptions

Exceptions can be handled robustly using the trycatch-finally syntax

- try: execute some code that may throw an exception
- If an exception is encountered, immediately start executing code in catch
- finally:
 - If no exception is thrown, execute after all code in the try block has executed
 - Execute after a return() in the try block before returning
 - If an exception is thrown, execute after all code in the catch block has executed

Handling Exceptions (cont.)

```
try{
  foo();
  thrower(); // might throw an exception
 bar();
catch (Exception e){
  caught();
finally {
 baz();
}
```

Throwing Exceptions

- Exceptions can be thrown using the *throw* statement:
 - throw exception
 - *exception* is an instance of an Exception class
- Example:

throw new IllegalArgumentException("Value of x should be greater than 5");

Exceptions can even be thrown from within a catch block

Catching Multiple Types of Exceptions

- A block of code may throw more than one type of exception
- Multiple catch blocks can be used to handle different types of exceptions that might be thrown from within a try block
- Only one catch block is executed per exception thrown within a try block
- The first catch block to match the thrown exception is the one executed

Catching Multiple Types of Exceptions (cont.)

```
try{
   thrower1(); // might throw ArithmeticException
   thrower2(); // might throw ArrayIndexOutOfBoundsException
}
catch(ArithmeticException e){
   foo();
}
catch(ArrayIndexOutOfBoundsException e){
   bar();
}
catch(Exception e){
   baz();
}
```

Utilizing Caught Exceptions

- When an exception is thrown, we may wish to identify where the exception originates, for debugging purposes
- We can use the following function call:

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

java.lang.IllegalArgumentException: Bad
parameter

- at Main.thrower(Main.java:99)
- at Main.exceptionTest(Main.java:105)
- at Main.main(<u>Main.java:125</u>)

Assertions

- An assertion is a statement that allows the programmer to test certain assumptions about some code
- If the assumption is correct, nothing happens
- Otherwise, an error is thrown
- Can be used to check preconditions and postconditions for some methods

Assertions (cont.)

- The syntax of assertions is as follows: assert *Expression*;
- *Expression* is some Java expression that returns a boolean value
- For example:

```
int i = 5;
String s = "Hello World";
...
assert i > 0;
assert s.equalsIgnoreCase("Hello World");
assert 1 == 2;
```

Assertions (cont.)

assert Expression;

- If *Expression* evaluates to true, nothing happens
- If *Expression* evaluates to false, an AssertionError is thrown

Errors vs. Exceptions

- Exceptions indicate an error condition that a reasonable application might want to handle
 - E.g.: a user-specified file is not found
- Errors indicate serious problems that a reasonable application should not try to handle
 - The Java virtual machine has run out of memory

Assertions (cont.)

Assertions throw Errors

- Thus, assertions should not be used to indicate problems within the scope of normal program execution
 - Assertions should indicate conditions that should never occur if the program is functioning properly
 - Assertions are generally removed from released software
- Exceptions should be used to indicate problems that can reasonably be expected to occur in various circumstances

Conventions for Using Assertions

 Don't use assertions to check public method preconditions (i.e., that parameters have acceptable values)

Instead: use exceptions

- You can use assertions to check nonpublic method preconditions
- You can use assertions to check postconditions on any method

Lab 8

- Modify your existing Project 3 code to add exception handling
- By the end of this lab, you should be able to:
 - 1. Create exception classes
 - 2. Throw exceptions to indicate errors
 - 3. Catch exceptions to yield robust behavior

Milestone 1

Create a FinchException class

- Extends Exception
- Contains a string that describes the error
- Implements toString()

Milestone 2

Throw FinchExceptions

- Must be thrown in response to illegal constructor parameters for all FinchActions
- FinchOrientationGuarded and FinchObstacleGuarded
 - If the orientation/obstacle type string is wrong, throw an exception
- Implementation should be straightforward

Milestone 3

Catch FinchExceptions

- Create a completely new driver class
- main() should attempt to create a set of FinchActions
 - Some constructor calls should contain illegal parameters, others shouldn't
- Report each thrown exception
- Add successfully created FinchActions to a FinchActionList

The General "To-Do"

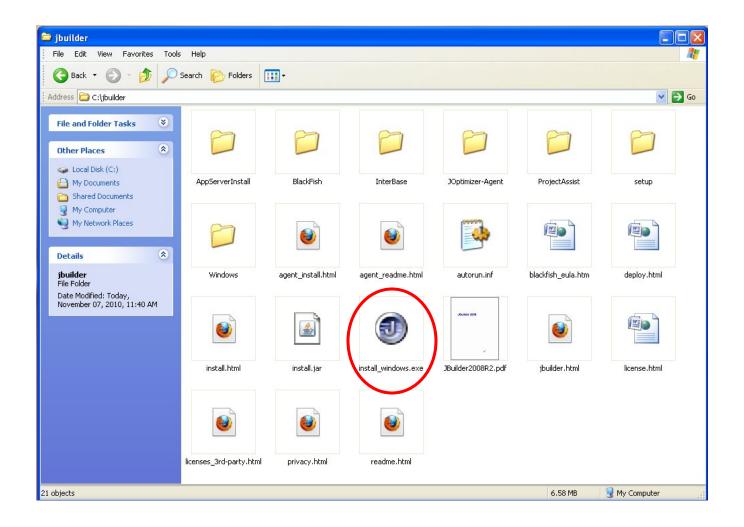
- Work on Lab 8 Today (avoid Project 4)
- Lab is due at 11:29 AM on Friday, November 12th
 - Demo
 - Electronic Submission

Testing...

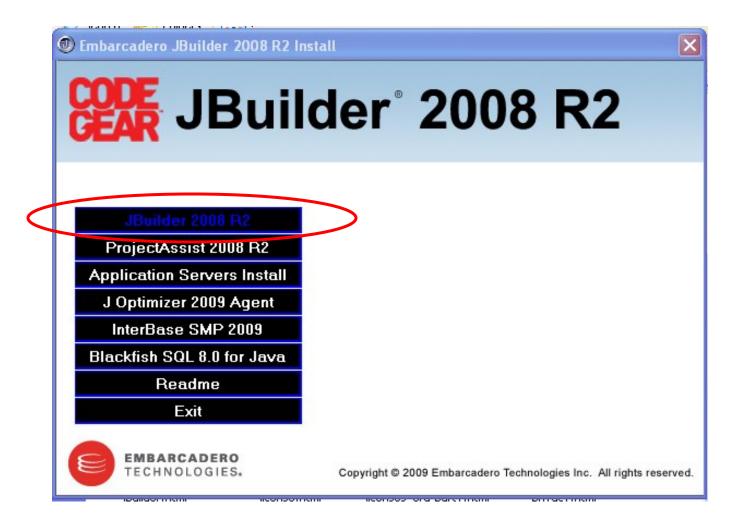
In the driver, for each FinchAction:

- Make a good constructor call
- Make a bad constructor call that does one of the following:
 - Too many/few parameters
 - Invalid orientation values
 - Invalid obstacle values
 - Invalid number formatting (if parsing strings)

Installing JBuilder



Choose JBuilder 2008 R2



Choose an install path

堰 JBuilder 2008 R2			
		Choose	Install Folder
 Introduction 	Where Would You Like to Install	?	
✓ License Agreement	C:\JBuilder2008R2		
→ Choose Install Folder		Restore Default Folder	Choose
 Choose Shortcut Folder Pre-Installation Summary Installing Install Complete 			
InstallAnywhere by Macrovision			
Cancel		Previous	Next

Choose where the shortcuts go

😼 JBuilder 2008 R2				
			Choose Sho	ortcut Folder
Introduction	Where would you like to create	product icons?		
License Agreement	🔘 In a new Program Group:	JBuilder 2008 R2		
 Choose Install Folder Choose Shortcut Folder 	O In an existing Program Group:	Accessories		~
O Pre-Installation Summary	💿 In the Start Menu			
 Installing Install Complete 	🔘 On the Desktop			
	🔘 In the Quick Launch Bar			
	Other:			Choose
	🔘 Don't create icons			
	Create Icons for All Users			
InstallAnywhere by Macrovision –			Draviaua	Next
Cancel			Previous	Next

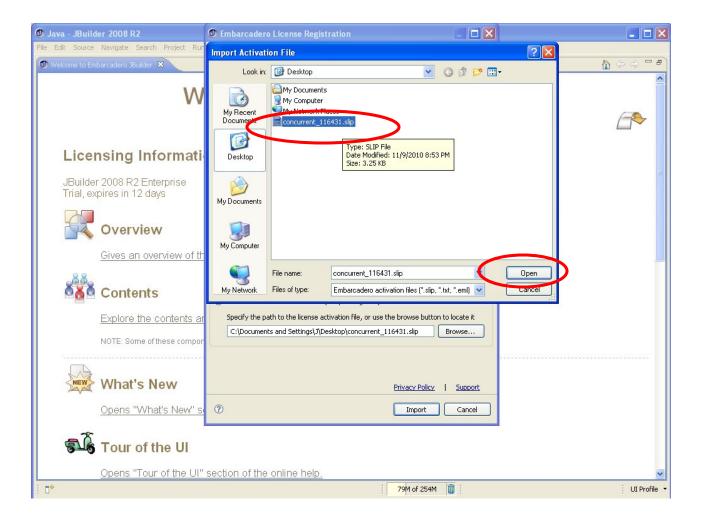
Select your workspace folder

() Worksp	ace Launcher	
	r orkspace 18 R2 stores your projects in a folder called a workspace. orkspace folder to use for this session.	
Workspace:	C:\Documents and Settings\J\JBuilder\workspace	Browse
Use this a	as the default and do not ask again	
$\mathbf{\nabla}$		OK Cancel

Choose to load an activation file

🗊 Java - JBuilder 2008 R2	Embarcadero Lice	nse Registration		
File Edit Source Navigate Search Project Run	JBuilder 2008 R2 Lic	ense Registration		
🐠 Welcome to Embarcadero JBuilder 🛛 🗙	🔕 Please specify a valid	activation file (*.slip or reg*.txt).		
W	Select one of the following	!		2
	O Register using online i	registration		
	Serial Number:			
Licensing Informati	Registration Code:	11931095		
JBuilder 2008 R2 Enterprise Trial, expires in 12 days	Developer Network A	kcount Details:		≡
	Login or Email:			
Overview	Password:	I need to create a Developer Networ	<u>'k(DN) account</u>	
Gives an overview of th	<u>I hav</u>	e a Developer Network (DN) account, but I've los	t my password	
Explore the contents		Trouble connecting? Try We tivation file (*.slip or reg*.txt) he license activation file, or use the browse buttor	\frown	
NOTE: Some of these pampor			Browse	/
What's New		Privacy Policy	<u>Support</u>	
Opens "What's New" s	0	Import	Cancel	
Tour of the UI				
Opens "Tour of the UI"	section of the onlin	<u>10 NOF 254M</u>	m :	UI Profile 🔻
: U'		79M of 254M	1	UI Profile 🔻

Select the slp license file



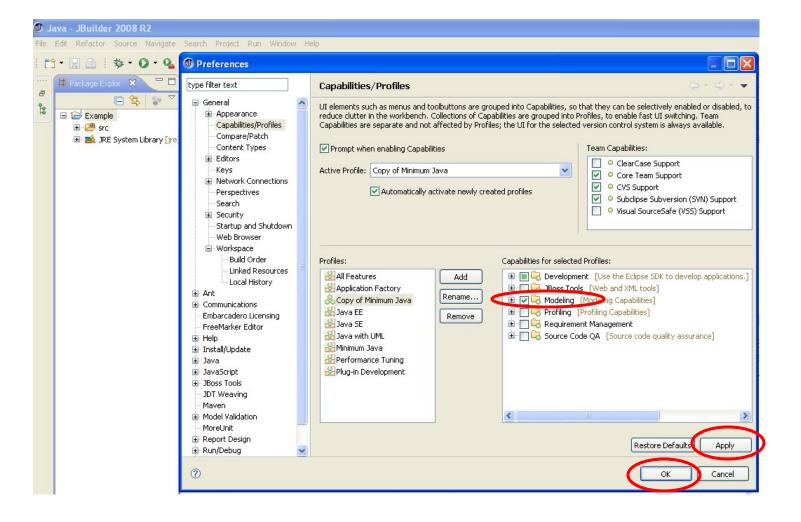
Finish up and restart JBuilder

🕑 Java - JBuilder 2008 R2	Imbarcadero License Registration	
File Edit Source Navigate Search Project Run	JBuilder 2008 R2 License Registration	
🜒 Welcome to Embarcadero JBuilder 🛛 🗙		
W	Select one of the following:	2
	O Register using online registration	
	Serial Number:	
Licensing Information	Registration Code: 11931095	
JBuilder 2008 R2 Enterprise Trial, expires in 12 days	Restart Required A restart is required for your changes to take effect. Do you want to restart	11
Overview	Ves No	
Gives an overview of th	Trouble connecting? Try Web Registration	-
Contents	● I have received an activation file (*.slip or reg*.txt)	
Explore the contents ar	Specify the path to the license activation file, or use the browse button to locate it	
NOTE: Some of these compor	C:\Documents and Settings\J\Desktop\concurrent_116431.slip Browse	
What's New	Privacy Policy Support	
Opens "What's New" s	⑦ Import Cancel	
Tour of the UI		-
Dep.	section of the online help.	×
i ≣ *	135M o ^F 254M	UI Profile 🔻

Create a new Java project (the same as using Eclipse)

		ect Run Window Help	
New Open File	Alt+Shift+N	🎽 Java Project	
Open rile		Projection	
Close	Ctrl+W	🖶 Package	o JBuilder 2008 R2
Close All	Ctrl+Shift+W	🞯 Class	J J J J J J J J J J J J J J J J J J J
J Save	Ctrl+S	🗊 Interface	
Save As		🕼 Enum	
🖹 Save All	Ctrl+Shift+S	🧼 Annotation	
Revert		🛱 Source Folder	
Move		🁆 Java Working Set	
Rename	F2	😂 Folder	
Refresh	F5	😭 File	
Convert Line Delimi	ters To	Untitled Text File	
Drint	Ctrl+P	Class by template	
	33317175	Report	
Switch Workspace		Template	26 12
Restart		🟦 Library	oducts.
🔄 Import		📑 JUnit Test Case	
👍 Export		💣 Task	
Properties	Alt+Enter	📑 Example	
Exit		📑 Other Ctrl+l	nts included.

Make sure modeling is enabled



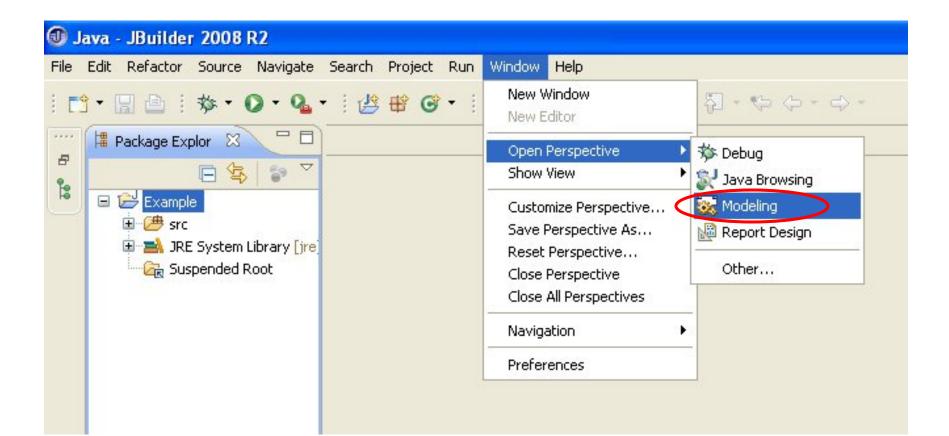
Right click on your project and "Enable Java Modeling"

Edit Refa	actor Source Navigate Search	n Project Run Window	He
• 🖬 d) 🎋 • 🔘 • 💁 • 🛃	3 🕸 🞯 • 🗄 🙆 🙋	9 6
Packa	ge Explor 🕅 🗖 🗖		
	Select on Diagram		
	New		۲
E 🔂 E	Go Inco		
±			
	Open Type Hierarchy	F4	
	Show In	Alt+Shift+W	•
	[Сору	Ctrl+C	
	Copy Qualified Name		
	💼 Paste	Ctrl+V	
	🔀 Delete	Delete	
	👲 Remove from Context	Ctrl+Alt+Shift+Dow	n
	Build Path		•
	Source	Alt+Shift+S	•
	Refactor	Alt+Shift+T	•
	🚵 Import		
	🛃 Export		
	🔗 Refresh	F5	
	Close Project		
	Close Unrelated Projects		
	Assign Working Sets		
	Add marker for tested meth	ods	
	Delete marker for tested me	thods	
	Run As		
	Debug As		•
	Profile As		
	Validate		_
C	Enable Java Modeling		

This window pops up (just hit ok)

Select classes and packag	es to be added to the model	
type filter text	Model Scope	⇔ + ⇔ + ▼
Model Scope	Enable model scope restriction for project Select packages and classes for modeling	
	密 🔳 🖶 (default package)	Select All Deselect All
	R	estore Defaults Apply
3		OK Cancel

Open the Modeling Perspective



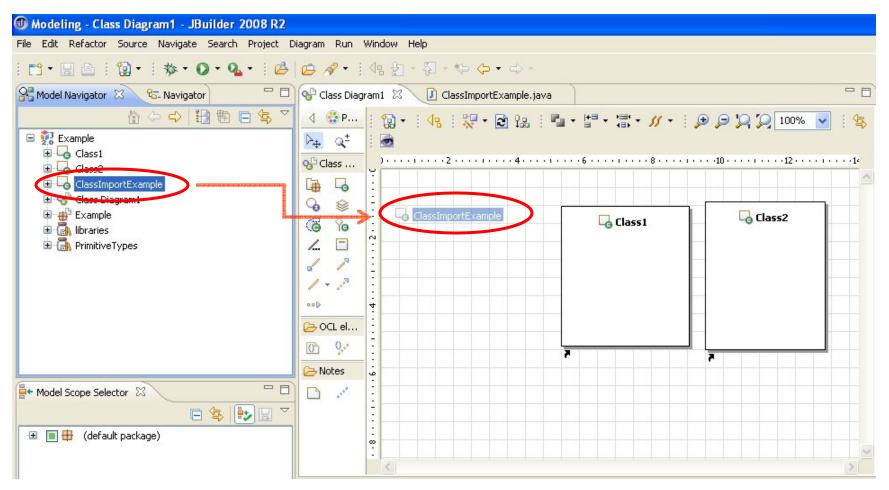
Create a New Class Diagram

1 Mode	eling - JBuilder 2008	R2	
File Edi	: Refactor Source Nav	igate Search Projec	t Diagram Run Window Help
: 📬 •	🖫 👜 i 🔞 • i 🕸	• 0 • 9 •	🗳 🖨 🛷 • 🕴 🔩 • 🖓 •
R Mod	el Navigator 🕴 😤 Na	vigator	
	🟠 🗇 🖕) 🖪 🔁 🗆 💲	
E 💱	Example		
2.0	New	Insert	•
	New Diagram		Activity
	Open	(😻 Class
	Open In Active Edito	r	E Component
	Open Type Hierarch		Composite Structure
	Show In	Alt+Shift+W	Geployment
	Paste		State Machine
			🖓 Use Case
	💢 Delete		
	Move		
	Rename		Sequence
	Refactor	Alt+Shift+T	▶ 1

Use the class tool to create new classes

	} } - २ - २ - २ - २ -	
VCD .		
😚 Class Diagram 1 🖇	×	
		■ - H - A - H - H - H - H - H - H - H - H
0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	,	
<		
Properties	ा Model Bookmarks 🖗 Serve	rs Problems
Properties	😚 Class Diagram1 sele	ected
Custom	Property	Value
	name full pame	Class Diagram1
Requirement	stereotype visibility metaclass	public Class Diagram
	App of the second s	Properties Properties Class Diagram1 sele Properties Custom Property name full name stereotype visibility

Drag existing classes onto the Class Diagram to add them



Tada!

