SECTION 2:

Loop Reasoning & HW3 Setup

cse331-staff@cs.washington.edu

slides borrowed and adapted from CSE 331 Spring 2018, CSE 391, and many more

Loop Invariants & Hoare Triples

- We can write a Hoare Triple involving a loop
 - {P} while(B) S {Q}
- The three key ingredients for a valid loop Hoare triple are:
 - The Invariant holds initially (precondition implies invariant)
 P => I
 - Loop body must re-establish the invariant (Inv holds each time we execute)
 - {I ∧ B} S {I}
 - Upon exiting the loop (test is false), the invariant must establish post-condition
 - $\{I \land !B\} => Q$

Review: Reasoning about loops

- What is a loop invariant?
 - An assertion that always holds at the top of a loop
- Why do we need invariants?
 - Most code is not straight line
 - Most programs aren't guaranteed to terminate
 - Therefore: We need invariants to prove the correctness of most programs we can encounter
 - Additionally, invariants help us write correct programs!

Loop Invariants ct.

- We want a goldilocks invariant
 - not too strong false and cannot be proven
 - not too weak cannot satisfy our postcondition
- No sure-fire way to find a loop invariant
 - Bad: Coding first and defining the invariant later
 - Good: think of invariant --> code the body --> code the loop condition --> code the initialization
- The common types of problems involving loop invariants include:
 - Given the code, fill in the assertions / invariant
 - Given a proof, find the error(s) in it if it is incorrect
 - Given the invariant, fill in the code

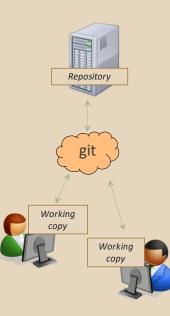
VERSION CONTROL

WHAT IS VERSION CONTROL?

- Also known as source control/revision control
- System for tracking changes to code
 - Software for developing software
- Essential for managing projects
 - See a history of changes
 - Revert back to an older version
 - Merge changes from multiple sources
- We'll be talking about git/GitLab, but there are alternatives
 - Subversion, Mercurial, CVS
 - Email, Dropbox, USB sticks (don't even think of doing this)

VERSION CONTROL ORGANIZATION

- A *repository* stores the master copy of the project
 - Someone creates the repo for a new project
 - \circ $\;$ Then nobody touches this copy directly $\;$
 - Lives on a server everyone can access
- Each person clones her own working copy
 - Makes a local copy of the repo
 - You'll always work off of this copy
 - The version control system syncs the repo and working copy (with your help)



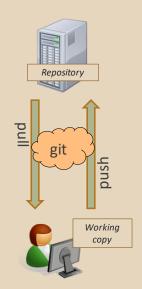
REPOSITORY

- Can create the repository anywhere
 - Can be on the same computer that you're going to work on, which might be ok for a personal project where you just want rollback protection
- But, usually you want the repository to be robust:
 - $\circ~$ On a computer that's up and running 24/7
 - Everyone always has access to the project
 - $\circ~$ On a computer that has a redundant file system
 - No more worries about that hard disk crash wiping away your project!
- We'll use CSE GitLab very similar to GitHub but tied to CSE accounts and authentication

VERSION CONTROL COMMON ACTIONS

Most common commands:

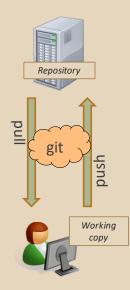
- add / commit / push
 - integrate changes *from* your working copy *into* the repository
- pull
 - integrate changes *into* your working copy *from* the repository



VERSION CONTROL UPDATING FILES

In a bit more detail:

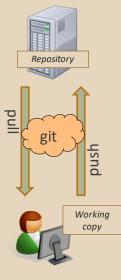
- You make some local changes, test them, etc., then...
- git add tell git which changed files you want to save in repo
- git commit save all files you've "add"ed in the local repo copy as an identifiable update
- git push synchronize with the GitLab repo by pushing local committed changes



VERSION CONTROL COMMON ACTIONS (CONT.)

Other common commands:

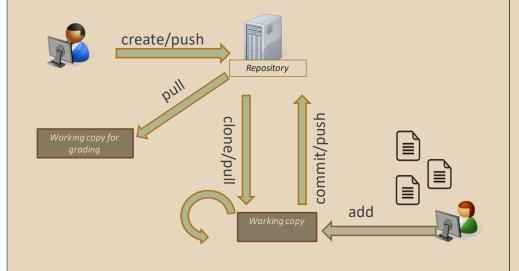
- add, rm
 - $\circ \ \ \, \mbox{add or delete a file in the working copy}$
 - just putting a new file in your working copy does not add it to the repo!
 - \circ $\;$ still need to commit to make permanent



THIS QUARTER

- We distribute starter code by adding it to your GitLab repo. You retrieve it with git clone the first time then git pull for later assignments
- You will write code using Eclipse
- You turn in your files by adding them to the repo, committing your changes, and eventually pushing accumulated changes to GitLab
- You "turn in" an assignment by tagging your repo and pushing the tag to GitLab
 - Do this after committing and pushing your files
- You will validate your homework by SSHing onto attu, cloning your repo, and running an Ant build file

331 VERSION CONTROL



Your Local Repository

LINKS TO DETAILED SETUP AND USAGE INSTRUCTIONS

- All References
 - https://courses.cs.washington.edu/courses/cse331/18su/#resources
- Machine Setup: Java, Eclipse, SSH
 - https://courses.cs.washington.edu/courses/cse331/18su/machine-setup.html
- Editing, Compiling, Running, and Testing Programs

 https://courses.cs.washington.edu/courses/cse331/18su/tools/editing-compiling.html
 https://courses.cs.washington.edu/courses/cse331/18su/tools/editing-compiling.html
 https://courses.cs.washington.edu/courses/cse331/18su/tools/editing-compiling.html
 https://courses.cs.washington.edu/courses/cse331/18su/tools/editing-compiling.html
 https://courses.cs.washington.edu/courses/cse331/18su/tools/editing-compiling.html
- Eclipse Reference
 - https://courses.cs.washington.edu/courses/cse331/18su/tools/eclipse_reference.html
- Version Control Git (includes setting up gitlab locally)
 - https://courses.cs.washington.edu/courses/cse331/18su/tools/versioncontrol.htm
 - <u>https://gitlab.cs.washington.edu/help/ssh/README.md</u>
- Assignment Submission
 - <u>https://courses.cs.washington.edu/courses/cse331/18su/tools/turnin.html</u>

DEVELOPER TOOLS

- Remote access
- Eclipse and Java versions
- Version Control

331 VERSION CONTROL

- Your main repository is on GitLab
- Only clone once (unless you're working in a lot of places)
- Don't forget to add/commit/push files!
 - Do this regularly for backup even before you're done!
- Check in your work!

Live Demo of Setup!

GIT BEST PRACTICES

- Add/commit/push your code EARLY and OFTEN!!!
 - You really, really, really don't want to deal with merge conflicts
 - Keep your repository up-to-date all the time
- Use the combined 'Commit and Push' tool in Eclipse
- Do not rename folders and files that we gave you – this will mess up our grading process and you could get a bad score
- Use the repo only for the homework
 - Adding other stuff (like notes from lecture) may mess up our grading process

HW 3

- Many small exercises to get you used to version control and tools and a Java refresher
- More information on homework instructions: <u>https://courses.cs.washington.edu/courses/cse3</u> <u>31/18su/hws/hw03/hw3.html</u>
- Cloning your repo: Instructions
- Committing changes: Instructions
 - How you turn in your assignments
- Updating changes: Instructions
 - How you retrieve new assignments

Turning in HW3

- Instructions
- Create a hw3-final tag on the last commit and push the tag to the repo (this can and should be done in Eclipse)
 - You can push a new hw3-final tag that overwrites the old one if you realize that you still need to make changes (Demo)
 - In Eclipse, just remember to check the correct checkboxes to overwrite existing tags
 - But keep track of how many late days you have left!
- After the final commit and tag pushed, remember to log on to attu and run ant validate

Turning in HW3

- Add/commit/push your final code
- Create a hw3-final tag on the last commit and push the tag to the repo (this can and should be done in Eclipse)
 - You can push a new hw3-final tag that overwrites the old one if you realize that you still need to make changes (Demo)
 - In Eclipse, just remember to check the correct checkboxes to overwrite existing tags
 - But keep track of how many late days you have left!
- After the final commit and tag pushed, remember to log on to attu and run ant validate

Ant Validate

- What will this do?
 - You start with a freshly cloned copy of your repo and do "git checkout hw3-final" to switch to the files you intend for us to grade, then run ant validate
 - Makes sure you have all the required files
 - Make sure your homework builds without errors
 - Passes specification and implementation tests in the repository
 - Note: this does not include the additional tests we will use when grading
 - This is just a sanity check that your current tests pass

Ant Validate

- How do you run ant validate?
 - Has to be done on attu from the command line since that is the environment your grading will be done on
 - Do not use the Eclipse ant validate build tool!
 - Be *sure* to use a fresh copy of your repo, and discard that copy when you're done
 - If you need to fix things, do it in your primary working copy (eclipse)

Ant Validate

- How do you run ant validate?
 - Steps
 - Log into attu via <u>SSH</u>
 - In attu, checkout a brand new local copy (clone) of your repository through the <u>command-line</u>
 - Note: Now, you have two local copies of your repository, one on your computer through Eclipse and one in attu
 - May need to create an SSH key on attu and add to GitLab: instructions
 - Go to the hw folder which you want to validate through the 'cd' command, then switch to the hw3 tag
 - For example: cd ~/cse331/src/hw3 git checkout hw3-final
 - Run ant validate

Ant Validate

- How do you know it works?
 - If successful, will output **Build Successful** at the bottom
 - If unsuccessful, will output **Build Failed** at the bottom with information on why
 - If ant validate failed, discard the validate copy of the repo on attu, fix and commit changes through eclipse, go back to attu, clone a fresh copy of the repo, and try ant validate again

WHAT IS ECLIPSE?

- Integrated development environment (IDE)
- Allows for software development from start to finish
 - Type code with syntax highlighting, warnings, etc.
 - Run code straight through or with breakpoints (debug)
 - Break code
- Mainly used for Java
 - Supports C, C++, JavaScript, PHP, Python, Ruby, etc.
- Alternatives
 - NetBeans, Visual Studio, IntelliJ IDEA

ECLIPSE

ECLIPSE SHORTCUTS

Shortcut	Purpose
Ctrl + D	Delete an entire line
Alt + Shift + R	Refactor (rename)
Ctrl + Shift + O	Clean up imports
Ctrl + /	Toggle comment
Ctrl + Shift + F	Make my code look nice 😳
Ctrl + Space	Autocomplete
Ctrl + S	Save (Eclipse does not autosave!)

ECLIPSE and Java

- Get Java 8
- Important: Java separates compile and execution, eg:
 - javac Example.java produces Example.class
 - Both compile and execute have to be the same Java!
- Please use Eclipse Oxygen, "Eclipse for Java Developers"
- Instructions: https://courses.cs.washington.edu/courses/cse331/18su/machine-setup.html#get-jdk

ECLIPSE DEBUGGING (if time)

- System.out.println() works for debugging...
 - It's quick
 - It's dirty
 - Everyone knows how to do it
- ...but there are drawbacks
 - What if I'm printing something that's null?
 - What if I want to look at something that can't easily be printed (e.g., what does my binary search tree look like now)?
- Eclipse's debugger is powerful...if you know how to use it

	• \$\$ \$\$ • \$ • 2	Quick Access	🔡 🖏 Java 🕸 De	ebug 🔠 🤮 SVN Repository	Exploring 🥐 PyDev	p [®] p [®]
Debug	1 23	200 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	🗇 😡= Variables 😒	© <mark>o Breakpoints</mark> ଶିଶ୍ରୀ Expre	ssions 🖄 🕫 🗄	~ - 0
	DelegatingMethodAccess	orImpl.invoke(Object, Object[])	lir A Name		Value	
	InvokeMethod.evaluate() 1 BlockJUnit4ClassRunner(P BlockJUnit4ClassRunner.ru BlockJUnit4ClassRunner.ru ParentRunner53.run() line:	ReflectiveCall() line: 45 lectiveCallable).run() line: 15 lexplosively(Object, Object) lin line: 20 arentRunner <t>).runLeaf(State unChild(GrameworkMethod, Ru unChild(Object, RunNotifier) lin 231</t>	em nN	117	RatPolyStackTes	t (id=33)
		arentRunner <t>).runChildren(</t>	Ru			
٠	ParentRunner<1>.access	000/ParentRunner_RunNotifier) i i			÷
) RatPol	lyStackTest.java 😒			8	Outline 🔀	
151 152 153 154 155⊕ 156 0157 158 159 160	//// Duplicate //// Duplicate public void testDupWithOme RatPolyStack stkl = star stkl.dup(); assertStackIs(stkl, "33); stkl = stack("123");	ck("3");			 testDivM testDivTi testDupV testDupV 	0 : void

Debug 🕄			🗱 Variables 🖾 🧕 Breakpoin	nts of Expressions	約 🕫 🗎	~ - (
	elegatingMethodAccessorIr ethod.invoke(Object, Objec	npl.invoke(Object, Object[]) lir +	Name		Value	
= Fr = Fr = In = BI = BI = Pa = Pa = BI = BI	ameworkMethod\$1.runRefl ameworkMethod\$1(Reflect ameworkMethod.invokeKry vokeMethod.evaluate() line ockUlnit4ClassRunner.runC ockUlnit4ClassRunner.runC ockUlnit4ClassRunner.runC ockUlnit4ClassRunner.fl.schedule(Ru ockUlnit4ClassRunner[Pare	ectiveCall() line: 45 iveCallable).run() line: 15 olosively(Object, Object) line: 20 ntRunner <t>).runLeaf(Statem hild(FrameworkMethod, Run)- hild(Object, RunNotifier) line: 1</t>	• this	III	RatPolyStackTest((d=33)
RatPolyStackTest	java 🛛			P 🗖 🔠 🗄 Outli	ine 🖾	- (
51 //////	(11111111111111111111111111111111111111		//////////////////////////////////////	/ 50	E B & XS	0 XL
53 / Do		0,	the left of your co e that the Java VN			id d re():v
	ring normal e	xecution of you	r program, and wa	it for acti	on from	ns() : ultVal

ECLIPSE DEBUGGING

Debug DelegatingMethodAc T Method.invoke(Obje FrameworkMethodS1	Click the Bug icon to mode. Otherwise yo won't stop at your b	ur program			p [■] p [●] (
FrameworkMethod51(%) FrameworkMethod51(%) FrameworkMethod.invo InvokeMethod.evaluate(BlockJUni4ClassRunner: BlockJUni4ClassRunner: ParentRunnef31.suchdlut BlockJUni4ClassRunner: BlockJUni4ClassRunner: ParentRunnef31.suchdlut BlockJUni4ClassRunner: ParentRunnef4T>.acces m	<	m) 	
RatPolyStackTestjava 23 151 152 153 154 155 155 156 157 158 159 159 154 155 155 156 157 158 158 159 159 150 151 152 153 154 155 155 156 157 158 159 159 150 151 152 153 154 155 156 157 158 159 150 151 152 153 154 155 156 157	ack("3"); 3");	- 1 1	01 01 000	testClear() testDiffere testDivMu testDivMu testDivTwv testDupWi testDupWi testDupWi testDupWi	void ntiate() : v ltiElems() : bElems() : thMultVal thOneVal(

ECLIPSE DEBUGGING

k • § • © ⇔ • ≤	Quick Access	🔡 🖓 Jav	Controllingyo	ur program	p ²
Method.invoke(Objec FrameworkMethodS1) FrameworkMethodS1) FrameworkMethod.in InvokeMethod.evalua BlockUnit4ClassRunn BlockUnit4ClassRunn	er(ParentRunner <t>).runLeaf(Statem er.runChild(FrameworkMethod, RunN er.runChild(Object, RunNotifier) line:</t>	e t	while debuggi with these but	-	3)
	dule(Runnable) line: 60 er(ParentRunner <t>).runChildren(Ru ess5000/ParentRunner_RunNotifier) li</t>		m		
🕽 RatPolyStackTest.java 🙁			- 0	BE Outline ⊠	
151 ////////////////////////////////////		,,,,,, ,, ,,,,,,,,,,,,,,,,,,,,,,,,,,,,		 E I k x e testClear(): x testClor(): v testDifferenti testDir/Wultil testDir/Wultil 	bid iate() : v Elems() : lems() : ·
<pre>159 assertStackIs(stk1, 160 stk1 = stack("123"); 161 stk1.dup(); 162 assertStackIs(stk1, 163 assertStackIs(stk1, 164 assertStackIs(stk1, 165 assertStackIs(stk1, 166 assertStackIs(stk1, 167 asse</pre>			•	 testDupWith testDupWith testDupWith 	OneVal(

Ø Debug ∷ ■ DeleastingMetho	i → P マロロ	Play, paus ⁽⁰⁾ = Variat Name		
Method.invoke((FrameworkMeth FrameworkMeth FrameworkMeth BrameworkMeth BrokeUnit4Class BlockUnit4Class BlockUnit4Class BlockUnit4Class BlockUnit4Class FlockUnit4Class BlockUnit4Class FlockUnit4Class Flock	Dbject. Diject) line: not available ddS1.runReflectiveCallbine: 45 odS1(ReflectiveCallbib).run() line: 15 od.invokeExplosively(Object, Object) line Runner(ParentRunner <t>).runLedf(Statem Runner.runChild(FramewordMethod, Runh Runner.runChild(Diget, RunNthefie) line:</t>	• this		RatPolyStackTest (id=33)
ParentRunnerS3.	run() line: 231 schedule(Runnable) line: 60	*	m	
	Runner(ParentRunner <t>).runChildren(Ru</t>			
ParentRunner <t iii<="" th=""><th>>.access\$000(ParentRunner_RunNotifier) li</th><th>4</th><th></th><th>)</th></t>	>.access\$000(ParentRunner_RunNotifier) li	4)
🗊 RatPolyStackTest.java 🙁		=	B	utline 🖾 😐
151 ///////////////////////////////////	//////////////////////////////////////		1.	 E [a] > >

to Debug		😭 🐉 Jav	Step into
<	DelegatingMethodAccessonImpLinvokc(Dipert, Object, Object)) inr - MethodLinvokc(Object, Object) line: not available FrameworkMethodSL:runReltectiveCall() line: 45 FrameworkMethodSL:runReltectiveCall() line: 45 FrameworkMethodLinvokeExplosively(Object, Object) line InvokeMethodLinvokeExplosively(Object, Object) line BlockUlnitkClassRunner:runChild(FrameworkMethod, Runh BlockUlnitkClassRunner:runChild(Dipert, RunNotifier) line: ParentRunner51.schedule(Runnable) line: 60 BlockUlnitkClassRunner(T>):runChildren(Ru ParentRunner51.schedule(Runnable) line: 60 BlockUlnitkClassRunner(T>):runChildren(Ru ParentRunner51.schedule(Runnable) line: 60 BlockUlnitkClassRunner(T>):runChildren(Ru ParentRunner51.schedule(Runnable) line: 60 BlockUlnitkClassRunner(T>):runChildren(Ru ParentRunner(T>):runChildren(Ru ParentRunner(T):runChildren(Ru ParentRunne	Name t	Steps into the method at the current execution point – if possible. If not possible then just proceeds to the next execution point. If there's multiple methods at the current execution
151 152 153 154 1550	yStackTestjava 🛿 ////////////////////////////////////	.	point step into the first one to be executed.
156	<pre>public void testDupWithOneVal() { RatPolyStack stk1 = stack("3"); stk1.dup();</pre>		testDivMultiElems() testDivTwoElems() testDupWithMultVal

ECLIPSE DEBUGGING

2 • 8	• ♥ ♥ • • • = Quick Access ::::::::::::::::::::::::::::::::::	(×)	Step Over	
	DelegatingMethodAccessorImpLinvoke(Object, Object[]) lir Method.invoke(Object, Object) line: not available FrameworkMethodS1:runReflectiveCall(] line: 45 FrameworkMethod.invokeExplosiveV(Object, Object) line: FrameworkMethod.invokeExplosiveV(Object, Object) line:	N	Steps over any method calls at the current execution point.	3)
	InvokeMethod.evaluate() line: 20 Block/Unit4ClassRunner(ParentRunner <t>).runLeaf(Statem Block/Unit4ClassRunner.runChild(TrameworkMethod, RunN) Block/Unit4ClassRunner.runChild(Object, RunNotifier) line: ParentRunner5J.schedule(Runnable) line: 60</t>	*	Theoretically program proceeds just to the next line.	
D-4D-1	Block/Unit4ClassRunner(ParentRunner <t>),runChildren(Ru ParentRunner<t>,access000/ParentRunner RunNotifier) is StackTest.iava 23</t></t>	*	BUT, if you have any breakpoints set that would be	,
151 152 153 154	Sackreugwo & //// Duplicate ////////////////////////////////////	1	hit in the method(s) you stepped over, execution will stop at those points instead.	4 . 0 : v
	<pre>public void testDupWithOneVal() { RatPolyStack stkl = stack("3"); stkl.dup(); assertStackIs(stkl, "33"); stkl = stack("123"); stkl.dup(); </pre>	Į	testDuryMathen testDuryMathen testDuryMathen testDuryMathen testDuryMathen testDuryMathen	ms() : ms() : lultVal neVal(

ECLIPSE DEBUGGING

			100	Step Out	
🌣 Debug 🖂		\$ ~ - -	(>)		
	DelegatingMethodAccessorimpLinvoke(Obje Methodinvoke(Object, Object) line not av FrameworkMethodS1.rnReflectiveCall() line FrameworkMethodS1.ReflectiveCallable).run FrameworkMethodLinvokeExplosively(Object InvokeMethod.evaluate() line: 20 BlockUlvnit4ClassRunner.runChild(Frameworl BlockUlvnit4ClassRunner.runChild(Object, Nameworl BlockUlvnit4ClassRunner.runChild(Object, Nameworl BlockUlvnit4ClassRunner.runChild(Object, Nameworl BlockUlvnit4ClassRunner.runChild(Object, Nameworl) BlockUlvnit4ClassRunner.runChild(Object, Nameworl) BlockUlvnit4ClassRunner.runChild(Dbject, Nameworl) BlockUlvnit4ClassRunner.runChild(Dbject, Nameworl) BlockUlvnit4ClassRunner.runChild(Dbject, Nameworl) BlockUlvnit4ClassRunner.runChild(Dbject, Nameworl)	ailable : 45 () line: 15 ; Object) line: :::::::::::::::::::::::::::::::::::	P	Allows method to finish and brings you up to the point where that method was called. Useful if you accidentally step	3)
< [ParentRunner53.run() line 231 ParentRunner51.schedule(Runnable) line: 60 Block/Unit4ClassRunner(ParentRunner(T>).r ParentRunner <t>.access5000/ParentRunner</t>		X	into Java internals (more on how to avoid this next).	•
	(Test.java 😒			Just like with step over though	
-		,,,,,,,,,,,,,,,,,,,,	1111		1
151 /// 152 /// 153 ///	//////////////////////////////////////		111 <u>1</u>	you may hit a breakpoint in the	4
151 /// 152 /// 153 /// 154 155⊕ @Te	//////////////////////////////////////		//// ////		0:v
151 /// 152 /// 153 /// 154 155⊕ @Te 156 pub			11A 1111	you may hit a breakpoint in the	0:v ns0: s0:-

	(⊜ • ⇒ • ≝ (⊕ • ⇒ • ≝	Quick Access	və 🖓 🗸	Enable/disable step filters
Preferences	Step Filtering		ites variat	There's a lot of code you don't
General Ant Code Recommenders Ant Code Recommenders Heig Install/Update Anos Code Style Code Style Code Style Code Style Debug Debug Legisla Structures Primitive Disalary OF Legisla Structures Primitive Disalary OF Legisla Structures Septements Septements Septements Subject Septements Subject Septements Subject Septements Subject Septements Subject Subjec	Step filters are applied when the Use Ste U use Step Filters Offend step filters Commission Commis	p Filter' toggle is activated. Add Filter	• t	want to enter when debugging, internals of Java, internals of JUnit, etc. You can skip these by configuring step filters. Checked items are skipped.
Properties Files Editor > Mayon Mylyn > PyDev > Raw/Debug > Team Validation > WindowBuilder > XM. <	Filter synthetic methods (requires VM Filter static inhisilars Filter constructors Filter simple getters Filter simple setters Step through filters	r support) Restore Defaults Apply OK Cancel	71 1 1111	IIIIIIIIIIIIII

🅸 Debug 💥 👘 😨 🖓 🗖 🗖	(x)= Variat	Stack Trace	
DelegatingMethodAccessorImpLinvoke(Object, Object]) lir Method.invoke(Object, Object) line: not available FrameworkMethod51.lentectiveCall(0) line: 45 FrameworkMethod51ReflectiveCallable).rung line: 15 FrameworkMethod.invokeExplosively(Object, Object) line: InvokeMethod.evaluate() line: 20 BlockUlnit4ClassRunner(ParentRunner(T>).runLeaf(Statem BlockUlnit4ClassRunner.Inclid(FrameworkMethod, Runh	Name © t	Shows what methods have been called to get you to current point where program is stopped.	3)
BlockUbnikClassRunner,runChild(Object, RunNotifier) line: ParentRunner53.run() line: 231 ParentRunner51.schedule(Runnable) line: 60 BlockUbnikClassRunner(ParentRunner <t>).runChildren(Ru ParentRunner<t>.accessS000/ParentRunner RunNotifier) li *</t></t>	•	You can click on different method names to navigate to that spot in the code	
⑦ RetPolyStackTestjava ಔ 551 ///////////////////////////////////	//////////////////////////////////////	without losing your current spot.	
<pre>154 155 @Test 155 mublic void testDupWithOneVal() { 0157 RatPolyStack stk1 = stack("3"); 158 stk1.dup(); 159 assertStackIs(stk1, "33"); 160 stk1.dup(); 161 stkl.dup(); 162 assertStackIs(stk1, "1123"); 162 assertStackIs(stk1, "1123"); 163 stk1.dup(); 164 stackIs(stk1, "1123"); 165 stk1.dup(); 166 stk1.dup(); 167 stk1.dup(); 168 stk1.dup(); 169 stk1.dup(); 160 stk1.dup(); 160 stk1.dup(); 160 stk1.dup(); 161 stk1.dup(); 162 stk1.dup(); 162 stk1.dup(); 163 stk1.dup(); 164 stk1.dup(); 165 stk1.dup(); 165 stk1.dup(); 166 stk1.dup(); 167 stk1.dup(); 168 stk1.dup(); 169 stk1.dup(); 169 stk1.dup(); 169 stk1.dup(); 169 stk1.dup(); 169 stk1.dup(); 160 stk1.dup(); 169 stk1.dup(); 160 stk1.dup(); 160 stk1.dup(); 160 stk1.dup(); 160 stk1.dup(); 160 stk1.dup(); 161 stk1.dup(); 1</pre>		testDifferential testDiff.woEle testDiv/twoEle testDupWithM testDupWithM testDupWithT testDupWithT	ems() : ms() : ultVal neVal(voVal(

ECLIPSE DEBUGGING

🔁 = 12 = 🗄 🕤 🕲 🕪 11 🛢 🖃 🕱 👁 12 🗟 🕱 12 🗟 🐨 🔍 🔍 🕸 = 🛛 = 🚱 😂 🔗 = 🖓 🖓 🖗 🗍 👘 👘

Variables Window			J Java 🕸 Debug 🔚 SVN Repository Exp	oloring PyDev	P P 3
		(x)=	Variables 🔀 💁 Breakpoints 🛱 Expressi	ons 🐇 🍓 📄	~
Chause all veriables including	ct[]) lir 🔺	Na	ime	Value	_
Shows all variables, including			this	RatPolyStackTest (id=33)
method parameters, local	5				
variables, and class variables,) line:				
· · · · ·	Statem				
	i, Runt-				
execution spot. Updates when	r) line:				
you change positions in the		•	m		÷.
stackframe. You can expand	ren(Ru				<u>^</u>
	ifier) li	٠			F.
objects to see child member		-			
values. There's a simple value		14/1		outline 🛛	
printed, but clicking on an item					A biou
will fill the box below the list	(11111111)	111	Some values are in t	ho form of	
):v
with a pretty format.			ObjectName (id=x), 1	his can be	s0 :
<pre>159 assertStackIs(stk1, "33");</pre>	J		used to tell if two va	riables are	tVal
160 stk1 = stack("123");			refering to the same	object	Val(Val(
<pre>161 stk1.dup(); 162 assertStackIs(stk1, "1123");</pre>			teleting to the same	object.	oid =

ECLIPSE DEBUGGING

	iables that have changed		🟦 👹 Java 🔅 Debug 🛗 SVN Repository Exploring	
sing	ce the last break point are	ct[]) lir 🔺	(x)= Variables 💥 💁 Breakpoints 🖓 Expression	Value
hig	hlighted in yellow.			
	- ,		▶ ● this	RatTermTest
		5	⊿ © t	RatTerm (id=
You	can change variables right) line:	⊳ <u>∎</u> coeff	RatNum (id=
	0 0	Statem	e ^r expt	5
troi	m this window by double	L Runh-		
clic	king the row entry in the	r) line:		
	0 ,			
Val	ue tab.			
-	 Diocioonnerciassicanner(Farencicanners 12).runci 	maren(Ru		
	= DecestPurpose To access \$000/DecestPurpose Purp			
•	ParentRunner <t>.access\$000/ParentRunner_Run</t>		-2*x^5	
<				M 08
-	m olyStackTest.java 🕄	Votifier) li 🔻	🖻 🖬 🗄 Outline (
 RatPc 151 152 	m olyStackTest.java 83	Votifier) li 🔻	🖻 🖬 🗄 Outline (tan N Ns @ NL ▽
151 152 153	m olyStackTest.java 🕄	Votifier) li 🔻	🖻 🖬 🗄 Outline (testClear(): void ▲
151 152 153 154	" blyStackTestjava 23 ////////////////////////////////////	Votifier) li 🔻	🖻 🖬 🗄 Outline (testClear(): void ∧ testClear(): void
151 152 153 154 155⊕	" b)StackTest.java 23 ////////////////////////////////////	Votifier) li 🔻	🖻 🖬 🗄 Outline (testClear() : void testClear() : void testCtor() : void testDifferentiate() : v
151 152 153 154 155⊖ 156	" blyStackTestjava 23 ////////////////////////////////////	Votifier) li 🔻	🖻 🖬 🗄 Outline (testClear() : void testClear() : void testDifferentiate() : v testDivMultiElems() : v
151 152 153 154 155 156 2157 158	<pre>m w w w w w w w w w w w w w w w w w w w</pre>	Votifier) li 🔻	🖻 🖬 🗄 Outline (testClear(): void testClear(): void testCtor(): void testDivfrentiate(): v testDivMultiElems(): testDivTwoElems():
151 152 153 154 155⊕ 156 257 158 159	<pre>"" " byStackTestjava 33 " " " " " " " " " " " " " " " " " "</pre>	Votifier) li 🔻	🖻 🖬 🗄 Outline (ing ing
151 152 153 154 155 156 90 157 158	<pre>m w w w w w w w w w w w w w w w w w w w</pre>	Votifier) li 🔻	🖻 🖬 🗄 Outline (testClear(): void testClear(): void testCtor(): void testDivfrentiate(): v testDivMultiElems(): testDivTwoElems():

ECLIPSE DEBUGGING

📑 • 🗄 • 🗒 🕤 🕑 💷 🔲 🧶 👁 . 🖉 🗮 🕱 🖉 🖉 🖉 • 🖉 • 🖉 • 🎯 🖉 • 🖗 🖉 • 🖉 🖉 • 🖉

ct[]) lir ^ 5) lines E Statem I, RunN- r) line:	Name		Value RatTermTest RatTerm (id= RatNum (id= 5
E Statem I, RunN			
rren(Ru tifier) li	-2*x^5		
		 testCtor testDiffe testDivM testDivT testDivT testDup 	
		(ffier) * -2*x^5	

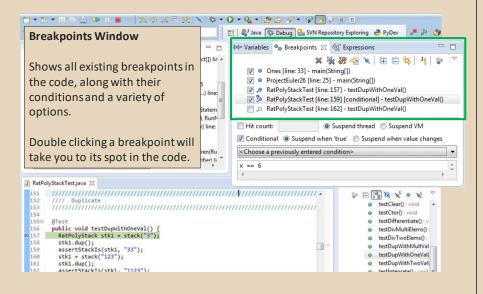
😂 + 😢 + 🗒 💩 🗇 म 🛢 👘 🔍 👁 🖉 🗮 🕱 🔍 🕸 + 🕗 + 💁 😂 🔗 + 🖓 🖓 🖗 🗐 👘

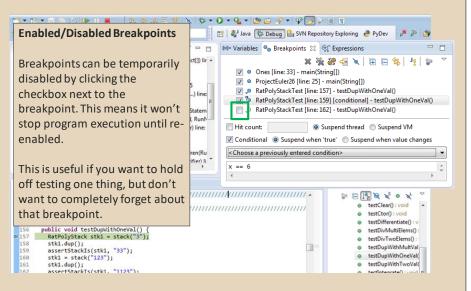
There's a powerful right-click menu.		😫 🐉 Java 🗱 Debug 🏭		sitory Exploring	v p ^s p ^s Ø
menu.	ct[]) lir 🔺	Name			Value
• See all references to a given		▷ ● this ▲ ● t			RatTermTest (id=33)
variable) line:	⊳ ∎ coeff ∎ expt		Select All Copy Variables	Ctrl+A Ctrl+C
See all instances of the	Statem			Find	Ctrl+F
variable's class	r) line:		-	Change Value	
Add watch statements for				All References All Instances	Ctrl+Shift+N
that variables value (more later)	ren(Ru ifier) li	-2*x^5	1	Instance Count New Detail Formatter Open Declared Type	Ctri+Sniit+N
151 ///////////////////////////////////		Aunner.class		Open Declared Type Hie	rarchy
154 1550 @Test 156 public void testDupWithOneVal() {				Instance Breakpoints Watch	
<pre>@157 RatPolyStack stk1 = stack("5"); 158 stk1.dup(); 169 assertStackIs(stk1, "33"); 161 stk1 = stack("123"); 161 stk1.dup(); 162 assertStackIs(stk1, "1123");</pre>			Q 1	 testDup 	Ctrl+Shift+I

ECLIPSE DEBUGGING

Show Logical Structure		🐉 Java 🏂 Debug 🔠 SVN Repository E	xploring 🧶 PyDev 🛛 🖻 🏂 🍠
0		🕪= Variables 🔀 💁 Breakpoints 🖧 🕯	Expressions 🐇 🔄 🔤
	ct[]) lir 🔺	Name	Value
Expands out list items so it's as		 this 	RatPolyStackTest (id=33)
if each list item were a field		⊿ © stk1	RatPolyStack (id=44)
	p) line:	⊿ ∎ ^E polys	Stack <e> (id=49)</e>
(and continues down for any	=	⊿ ▲ [0]	RatPoly (id=719)
children list items)	Statem	a 🧉 terms	ArrayList <e> (id=728)</e>
ciliaren iist itemisj	i, Runt	a 🔺 [0]	RatTerm (id=731)
BlockJUnit4ClassRunner.runChild(Object, RunN	otifier) line:	⊳ 🖬 coeff	RatNum (id=733)
ParentRunner\$3.run() line: 231		expt	0
ParentRunner\$1.schedule(Runnable) line: 60			
BlockJUnit4ClassRunner(ParentRunner <t>).run</t>			
	The Distant Age	3	
ParentRunner <t>.access\$000/ParentRunner_Ru</t>	inNotifier) li 🍷	3	
<	inNotifier) li *	3	
الله المعامل ا المعامل المعامل	•	•	
<	•	-	₽ □ □ 0 × ×
<	•	•	Image: The second s
<	•	•	
m D RatPolyStackTest.java 23 151 ///// Duplicate 153 ////////////////////////////////////	•	•	testClear() : void testCtor() : void testCtor() : void testDifferentiate() : v
< RatPolyStackTestjava 23 15.1 ///////////////////////////////////	•	•	 testClear(): void testCtor(): void testDifferentiate(): v testDivMultiElems():
<	•	•	testClear():void testCtor():void testDifferentiate():v testDifferentiate():v testDivfluitElems(): testDivfluitElems():
<pre></pre>	•	•	testClear(): void testCtor(): void testCtor(): void testDifferentiate(): v testDivfMultiElems(): testDivfMultiElems(): testDupWithMultVal
<	•	•	testClear():void testCtor():void testDifferentiate():v testDifferentiate():v testDivfluitElems(): testDivfluitElems():

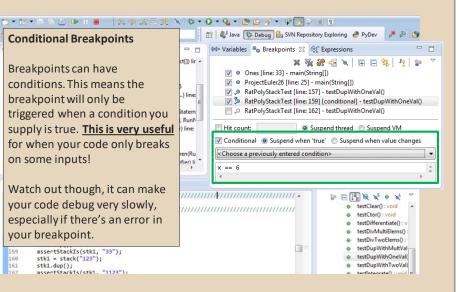
ECLIPSE DEBUGGING





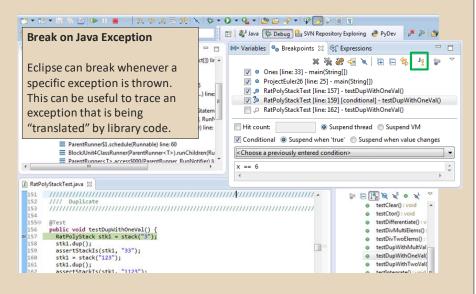
≫ Debug ⊠ 🙀 ⇒ 😜 🌣	- 0	🗱= Variables 💁 Breakpoints 🔀 🖓 Expressions	-
E DelegatingMethodAccessorImpLinvoke(Object, Object, Obj	ct[]) lir + 5) line: E Statem I, RunN r) line:	Image: State	VithOneVal()
	ren(Ru ifier) li *	Conditional Suspend when 'true' Suspend when var Choose a previously entered condition> x == 6 (lue changes
<pre>Win code execution stop. 153 154 1559 155 155 155 155 155 155 155 155 15</pre>		Clear) Clear)	roid tiate():v Elems(): elems(): MultVal iOneVal(iTwoVal(

ECLIPSE DEBUGGING



ECLIPSE DEBUGGING

Disable All Breakpoints		🖹 🐉 Java 🏂 Debug) 🛗 SVN Repository Exploring 🛛 🥚 P	yDev 👂 🔊 🎒		
		🗱 Variables 💁 Breakpoints 💥 🖓 Expressions			
	ct[]) lir 🔺	x 💥 🌮 🚽 🔍 🗉	E 🔄 🧏 😨		
You can disable all breakpoint	S	Ø Ones [line: 33] - main(String[])			
temporarily. This is useful if	5 📼	ProjectEuler26 [line: 25] - main(String[])			
' ') line:	RatPolyStackTest [line: 157] - testDupWithO	-		
you've identified a bug in the	E	V 3 RatPolyStackTest [line: 159] [conditional] - testDupWithOneVal()			
middle of a run but want to le	t Statem	P RatPolyStack est [line: 162] - testDupWithO	t [line: 162] - testDupWithOneVal()		
the rest of the run finish	r) line:	Hit count: Suspend thread	Suspend VM		
		Conditional Suspend when 'true' Suspend	when value changes		
normally					
normany.	(D	(colores a sectional section)	1		
normany.	ren(Ru ifier) li *	Choose a previously entered condition>]		
		<choose a="" condition="" entered="" previously=""> x == 6</choose>			
Don't forget to re-enable	ifier) li 🍷	<u></u>	4		
Don't forget to re-enable	ifier) li *	x == 6 4) >>> o xL ⊽		
Don't forget to re-enable breakpoints when you want to	ifier) li *	x == 6	k k k k k k k k k k k k k k		
Don't forget to re-enable breakpoints when you want to use them again.	ifier) li *	x == 6 ∢ ///////////////////////////////////	R X ^S ● X ^L ▽ tClear() : void ^ tCtor() : void		
Don't forget to re-enable breakpoints when you want to use them again.	ifier) li *	x == 6 < /////////////////////////////////	tCtor() : void tDifferentiate() : v		
Don't forget to re-enable breakpoints when you want to use them again.	ifier) li *	x == 6 <	tCtor() : void tDifferentiate() : v tDivMultiElems() :		
Don't forget to re-enable breakpoints when you want to use them again.	ifier) li *	x == 6 x y y y y y y y y y y y y y y y y y y	tCtor() : void tDifferentiate() : v		
<pre>156 public void testDupWithOneVal() { 157 RatPolyStack stk1 = stack("3");</pre>	ifier) li *	x == 6 4	tCtor() : void tDifferentiate() : v tDivMultiElems() : tDivTwoElems() :		



Expressions Window

Used to show the results of custom expressions you provide, and can change any time.

Not shown by default but highly recommended.

/indow Help		
New Window New Editor Hide Toolbar	ジ ひ 回 前 タ マ マ や 体 梦 Debug 晶 SVN Repository Explorin) •
Open Perspective	akpoints of Expressions 🔀	
Show View	条 Ant	
Customize Perspective Save Perspective As Reset Perspective Close Perspective Close All Perspectives	Breakpoints Alt+Shift+Q, B Console Alt+Shift+Q, C Debug Display Image: Display Int+Shift+Q, L Image: Display Alt+Shift+Q, L Image: Display Alt+Shift+Q, L	
Navigation Preferences	Coutline Alt+Shift+Q, O Tasks	
hash32	(X)= Variables Alt+Shift+Q, V	
	Other Alt+Shift+Q, Q	

ECLIPSE DEBUGGING

Expressions Window

Used to show the results of custom expressions you provide, and can change any time.

Resolves variables, allows method calls, even arbitrary statements "2+2"

Beware method calls that mutate program state – e.g. stk1.clear() or in.nextLine() – these take effect immediately

stk1.dup();
assertStackIs(stk1, "33");

stk1.dup();
assertStackTs(stk1. "1123");

stk1 = stack("123");

159

160

- -🗱 Variables 💁 Breakpoints 🙀 Expressions 🔀 (]) lir + 🗄 🍕 🖻 🛉 ¥ 🔆 ▲ [3, 2, 1, null, nul Name Value *** "this" (id=33) line: (id=57) ▲ ^{X+Y} "stk1.polys" (id=61) atem capacityIncrement 0 Runt elementCount 3 line: b 🔶 elementData Object[10] (id=73) o modCount 3 stk1.toString() hw4.RatPolyStack(n(Ru ier) li hash 0 hash32 0 4 ► **∢** P □ 1 2 2 2 2 0 2^L e testClear() : void * testCtor(): void testDifferentiate() : v testDivMultiElems(): testDivTwoElems(): testDupWithMultVal testDupWithOneVal(testDupWithTwoVal((http://terratel

🌣 • 🜔 • 💁 • 🎥 🗁 🔗 • 🖓 🍠 🖗 🗍 🖷

👔 🔡 Java 🗱 Debug 🛗 SVN Repository Exploring PyDev

. 2 3

Expressions Window		🛓 👻 🗁 🥪 🗣 🖓 🍹	itory Exploring PyDev	
These persist across projects, so clear out old ones as necessary.	(]) lir 🔺 Nar		Value	£ ↔ □ + ¥ 3
Frameworknemossjytenetweckamability.00 FrameworkNethod.invokeExplosively(Object, C InvokeMethod.evaluate() line: 20 BiockUlnik(LassRunner(TarenRunner <t>).rur BiockUlnik(LassRunner.unchild(Cabject, RunN ParentRunnerS3.schedule(Runnable) line: 60 BiockUlnik(LassRunner.t).increase(S000)/ParentRunner.Runner(T>).rur ParentRunner(T>.accessS000)/ParentRunner.Ru</t>	Dbject) line E A hLeaf(Statem Method, RunN Notifier) line: hChildren(Ru	<pre>% "this" % "stk1.polys"</pre>	(id=33) (id=57) (id=61) 0 3 Object[10] (id=73) 3 hw4.RatPolyStack(0 0	
P RatPolyStackTest.java & 151 /////////////////////////////////	· · · · · · · · · · · · · · · · · · ·	m ///////*	× ∞ 🖓 🖉 v	<
<pre>152 //// Duplicate 153 //// Duplicate 154 154 155 @Test 156 public void testDupMithOneVal() { 158 stkl.dup(); 159 assertStack(stkl = stack("3"); 169 assertStack("123"); 161 stkl = stack("123"); 161 assertStack(s(stkl, "1123"); 162 assertStack(s(stkl, "1123"); 163 assertStack(s(stkl, "1123"); 164 assertStack(s(stkl, "1123"); 165 assertStack(s(stkl, "1123"); 165 assertStack(s(stkl, "1123"); 166 assertStack(s(stkl, "1123"); 167 assertStack(s(stkl, "1123"); 168 assertStack(s(stkl, "1123"); 169 assertStack(s(stkl, "1123"); 160 assertStack(s(stkl, "1123"); 161 assertStack(s(stkl, "1123");</pre>			testCtor testDirf testDirf testDirf testDup testDup testDup testDup	r() : void rentiate() : v JultiElems() : WithOutVal WithOneVal(WithTwoVal(osto) : void =