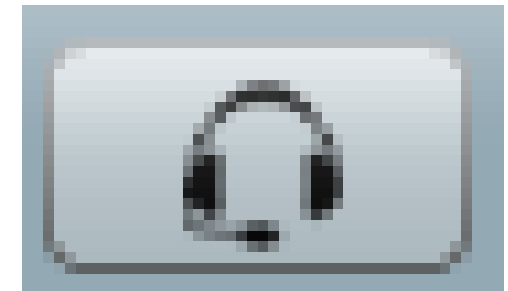
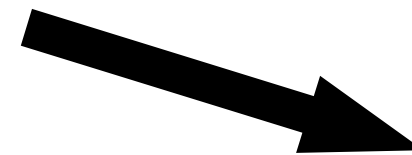


# Welcome to the JavaWIDE Webinar

1. Close all windows except web browser
2. Look for JavaWIDE on right side of screen
3. Turn your sound on Upper left side of screen
4. Listen to music, wait for 8pm



# Pre-Webinar Orientation

to  **BigBlueButton**

1. Overview of the Environment
2. Turning on your Audio
3. Using the Chat Window
4. Controlling Desktop Sharing
5. Using Full Screen Frames

# Webinar Environment

Presentation

JavaWIDE

Audio

Demo Desktop

Movable Divider

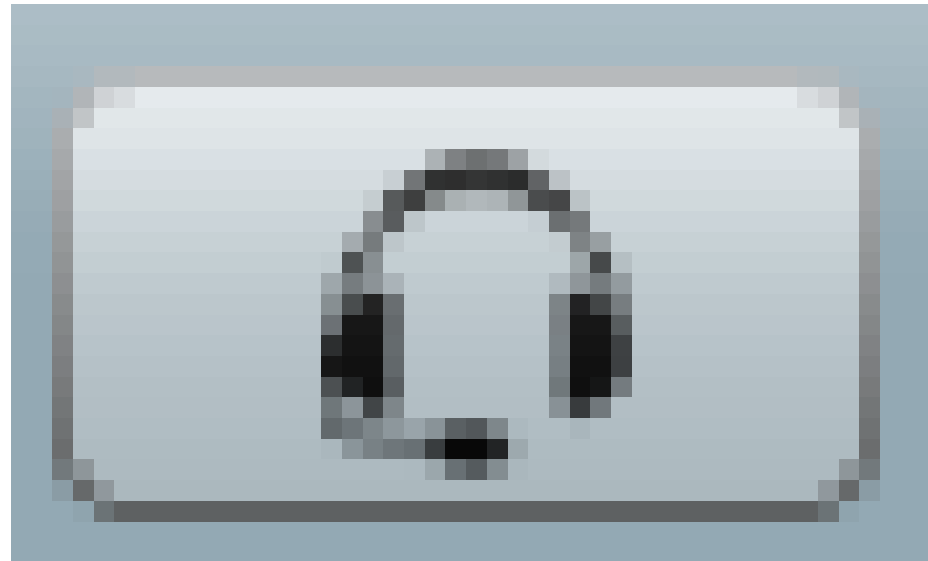
Chat

The screenshot shows the JavaWIDE interface with several components:

- Users:** A table with columns 'Role' and 'Name'. It lists 'great (you)'.
- Chat:** A chat window titled 'CSTAWebinar' with a message: 'Welcome to Jam Jenkins's meeting.' and a 'Send' button.
- Presentation:** A window titled 'Java Wiki' containing the text: 'Java WIDE Integrated Development Environment', 'Dr. Jam Jenkins, Assistant Professor Information Technology, Georgia Gwinnett College cjenkins@ggc.usg.edu, http://www.jamjenkins.org', and 'Computer Science Teachers Association Webinar 8:00pm-9:00pm October 19, 2010'.
- Desktop Sharing:** A window showing a desktop environment with a 'Welcome to JavaWIDE!' dialog box. The dialog has options: 'Help me get started.', 'Create a new program.', 'Open an existing program.', and 'Log in.'.
- JavaWIDE Welcome Panel:** A vertical panel on the right with the title 'Welcome to JavaWIDE!' and the question 'What would you like to do?'. It contains four buttons: 'Help me get started.', 'Create a new progr...', 'Open an existing pr...', and 'Log in.'.

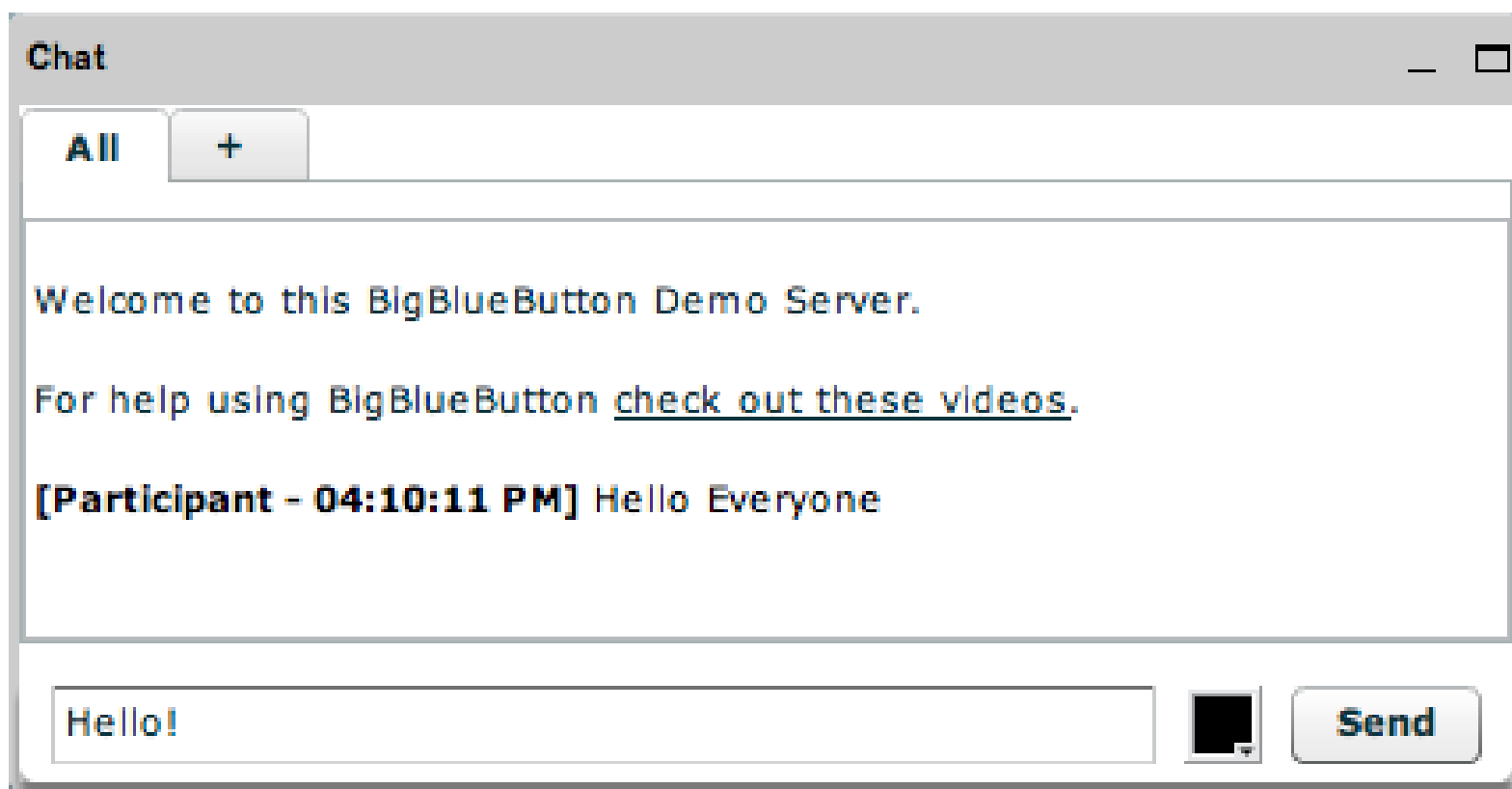
# Audio

- Click on Headset Icon on the top left side of screen to listen
- Use chat to ask questions



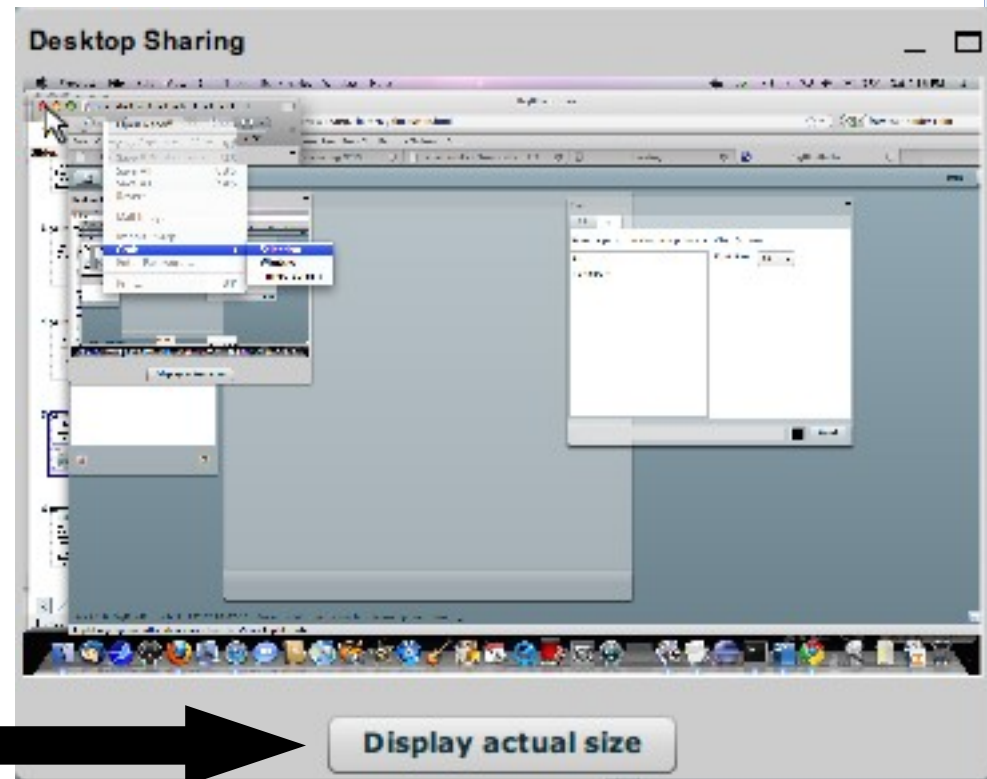
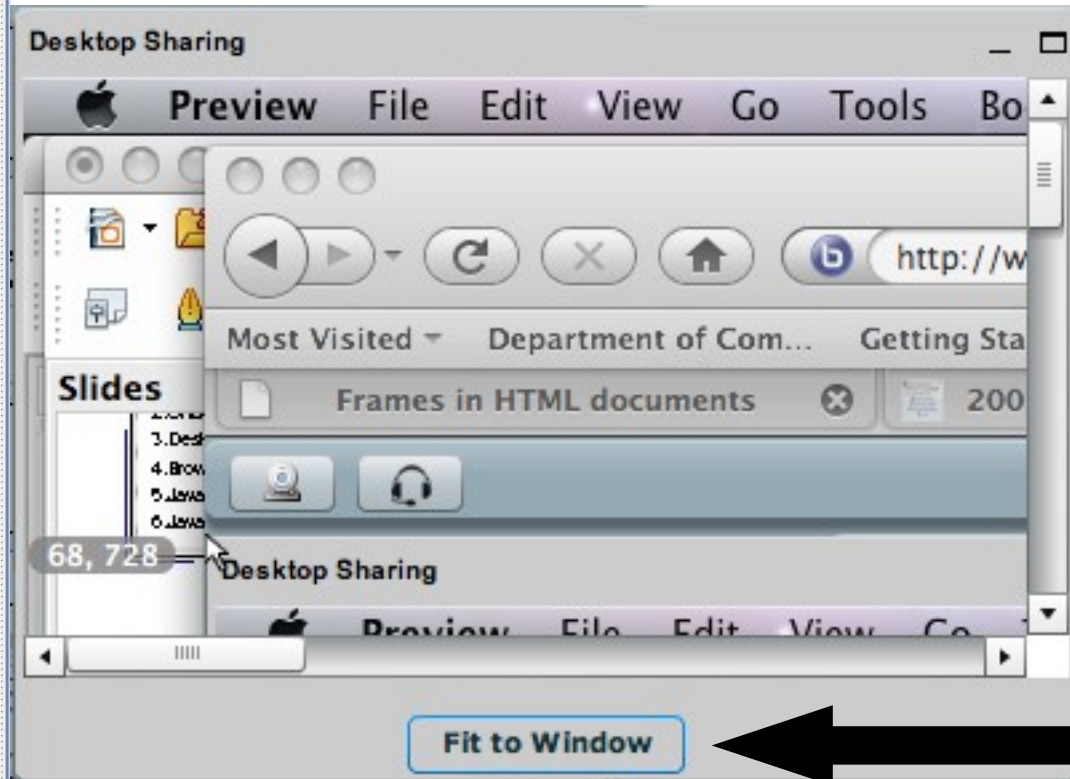
# Chat Window

- Say hello in the chat window



# Desktop Sharing

- Toggle between two modes: Fit in Window & Actual Size

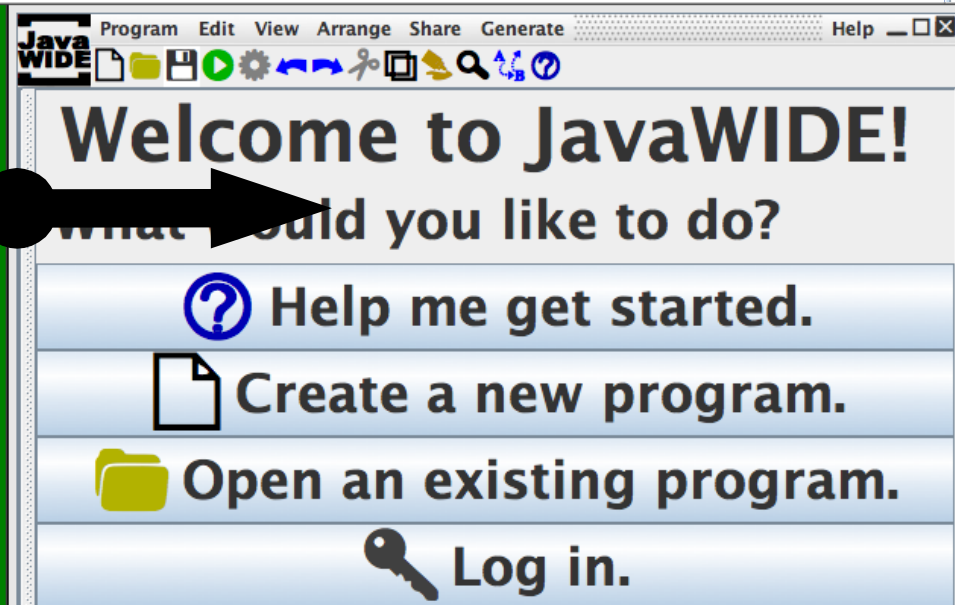
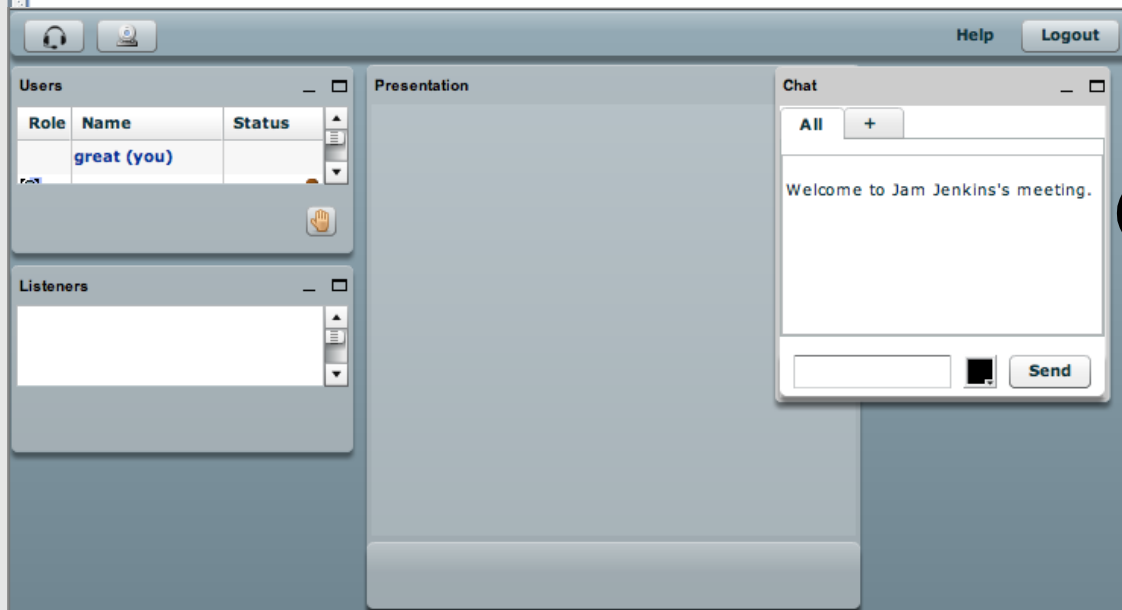


# Full Screen/Frames

Browser Menu:

View → Full Screen

↔ **Green Slider** ↔



# Webinar Environment

Presentation

JavaWIDE

Audio

Demo Desktop

Movable Divider

Chat

The screenshot shows the JavaWIDE interface with several components:

- Users:** A table with columns 'Role' and 'Name'. It lists 'great (you)'.
- Chat:** A chat window titled 'CSTAWebinar' with a message: 'Welcome to Jam Jenkins's meeting.' and a 'Send' button.
- Presentation:** A window titled 'Java Wiki' containing the text: 'Java Wiki Integrated Development Environment', 'Dr. Jam Jenkins, Assistant Professor Information Technology, Georgia Gwinnett College cjenkins@ggc.usg.edu, http://www.jamjenkins.org', and 'Computer Science Teachers Association Webinar 8:00pm-9:00pm October 19, 2010'.
- Desktop Sharing:** A window showing a desktop environment with a 'Welcome to JavaWIDE!' dialog box. The dialog has options: 'Help me get started.', 'Create a new program.', 'Open an existing program.', and 'Log in.'.
- JavaWIDE Welcome Panel:** A vertical panel on the right with the title 'Welcome to JavaWIDE!' and the question 'What would you like to do?'. It contains four buttons: 'Help me get started.', 'Create a new progr...', 'Open an existing pr...', and 'Log in.'.

# Java Wiki



# Integrated Development Environment

Dr. Jam Jenkins, Assistant Professor  
Information Technology, Georgia Gwinnett College  
[cjenkins@ggc.usg.edu](mailto:cjenkins@ggc.usg.edu), <http://www.jamjenkins.org>

JavaWIDE Site Owners Webinar  
8:00pm-9:00pm November 2, 2010

# Outline of Webinar

- History/Overview
- Pedagogical Motivation
- Interactive Demonstrations
- Additional Resources
- Conclusion & Questions

# Brief History

First Prototype: December 2007

First College Class: January 2008

First Used in India: Summer 2008

First Online Course: Summer 2009

First H.S. Adoption: August 2009

Concurrent Editing: November 2009

50+ JavaWIDE Sites: October 2010

Digital Ink Added: November 2010

# U.S. JavaWIDE Sites\*



\* Sign up for a FREE JavaWIDE site at the end of this webinar.

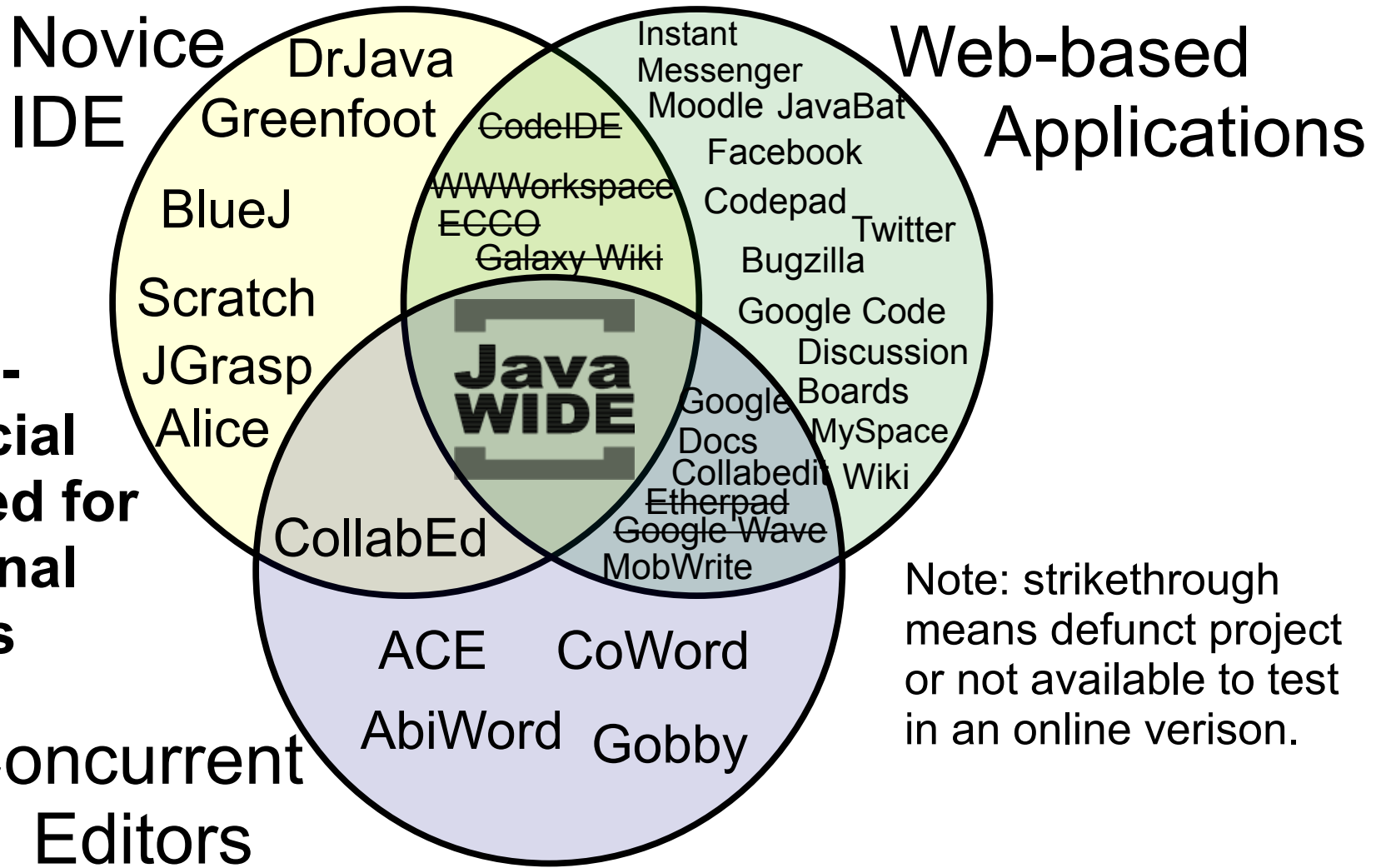
# JavaWIDE is ...

- **Java**  
AP Computer Science Language
- **Wiki**  
Good for Sharing & Collaboration
- **Integrated**  
One Tool with Many Parts
- **Development Environment**  
Good Programming Support

# Innovative

- Entirely browser based applet IDE
- Realtime group programming
- Novice friendly simple interface
- Version control, auto-import, code completion, embedded API, syntax highlighting, auto-indent, social network integration, GUIs, ...

# Unique



**Free non-commercial tools used for educational purposes**

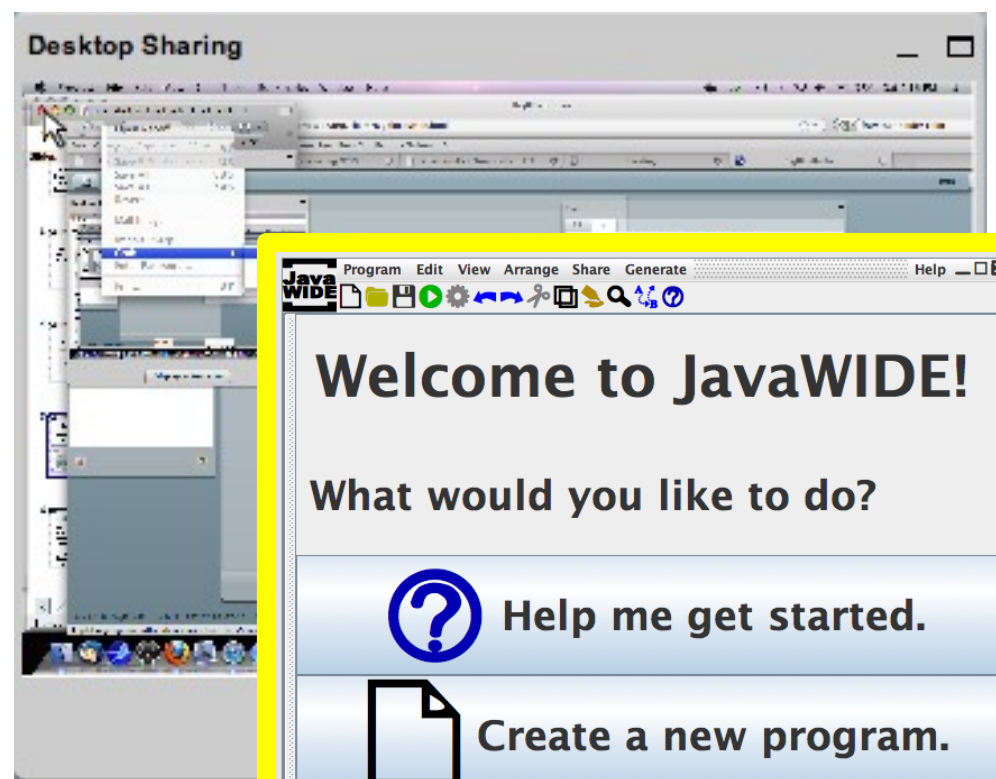
Note: strikethrough means defunct project or not available to test in an online version.

# Pedagogy

- IDE available on all computers
- Simple, easy interface
- Example code with revisions
- Interactive code examples
- Group work space
- No lost work – stored on server

# Demo Format

1. Overview/  
Motivation
2. Watch Demo
3. Participate



# Detach IDE View

- *Motivation*

Maximize Work Space

- *Watch Demo*

Desktop Sharing



# Participate

- Click on title bar to detach
- Click on X to reattach
- Resize from edges

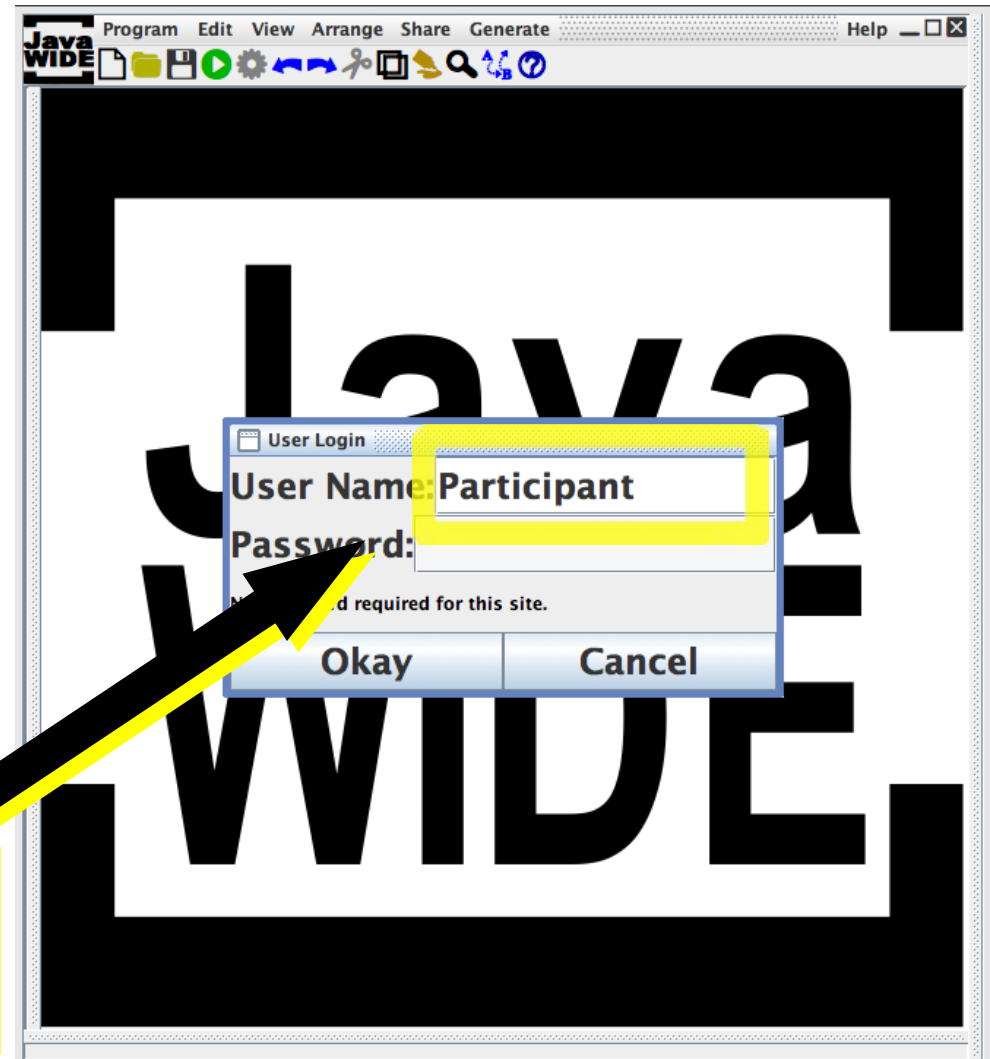


# Login Demo

- *Motivation*  
Identify Users
  
- *Watch Demo*  
Desktop Sharing



# Please Login



# Follow Demo

- *Motivation*  
**Synchronize Open Programs**
  
- *Watch Demo*  
**Desktop Sharing**



# View → Follow → Teacher

The image shows two side-by-side screenshots of the JavaWIDE web application. The left screenshot displays the main interface with a 'View' menu open, highlighting the 'Follow' option and its sub-item 'Teacher'. A yellow box highlights the 'View' menu, and a black arrow points from it to the right screenshot. The right screenshot shows a code editor with the following Java code:

```
1 package teacher;  
2  
3  
4 /**  
5  * All about my application.  
6  * @author Teacher  
7  */  
8 public class FirstProgram  
9 {  
10     public static void main(String[] args)  
11     {  
12         System.out.println("Hello JavaWIDE!");  
13     }  
14 }
```

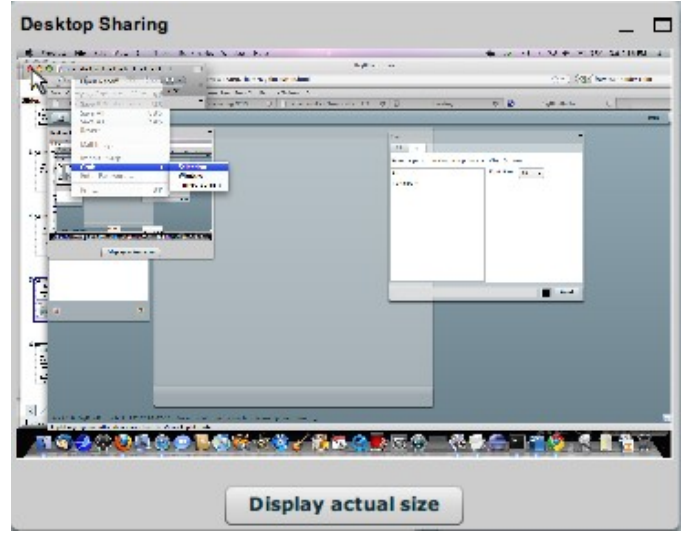
# Edit → Save → Run Demo

- *Motivation*

Modify, Store, and Execute

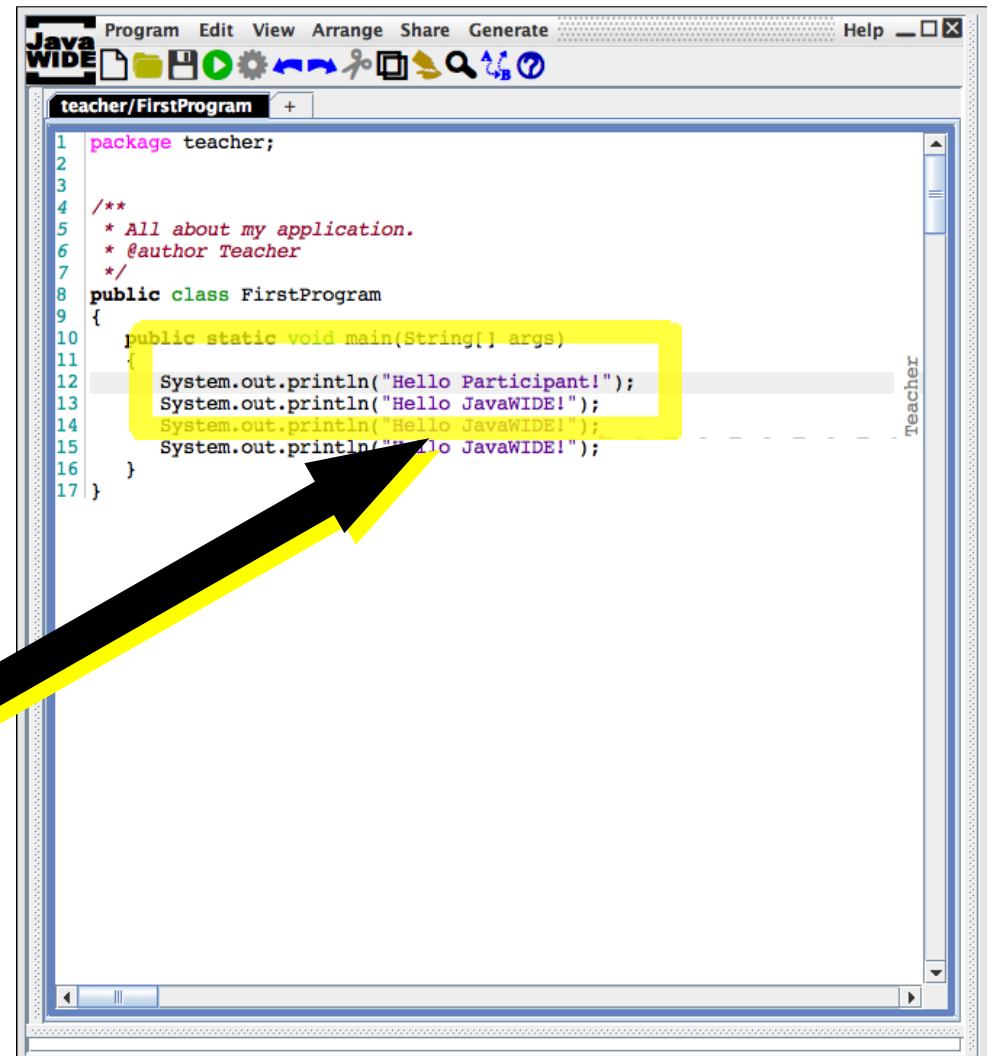
- *Watch Demo*

Desktop Sharing



# Print Your Name

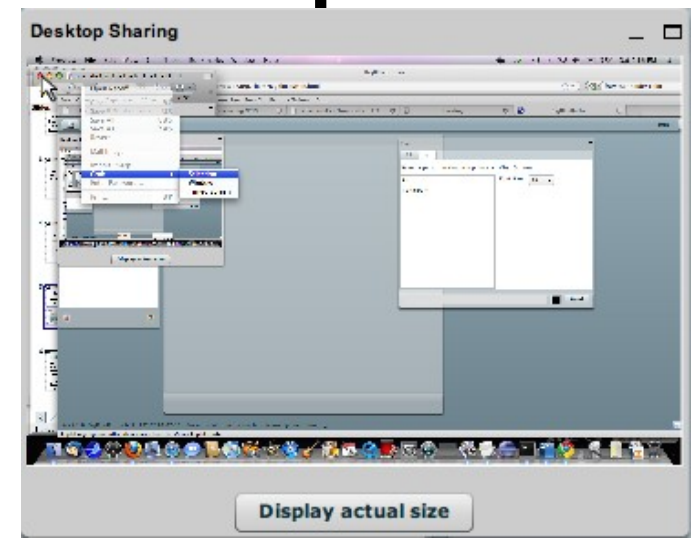
- Put cursor on a line of its own (not near anyone else)
- Change to print your name



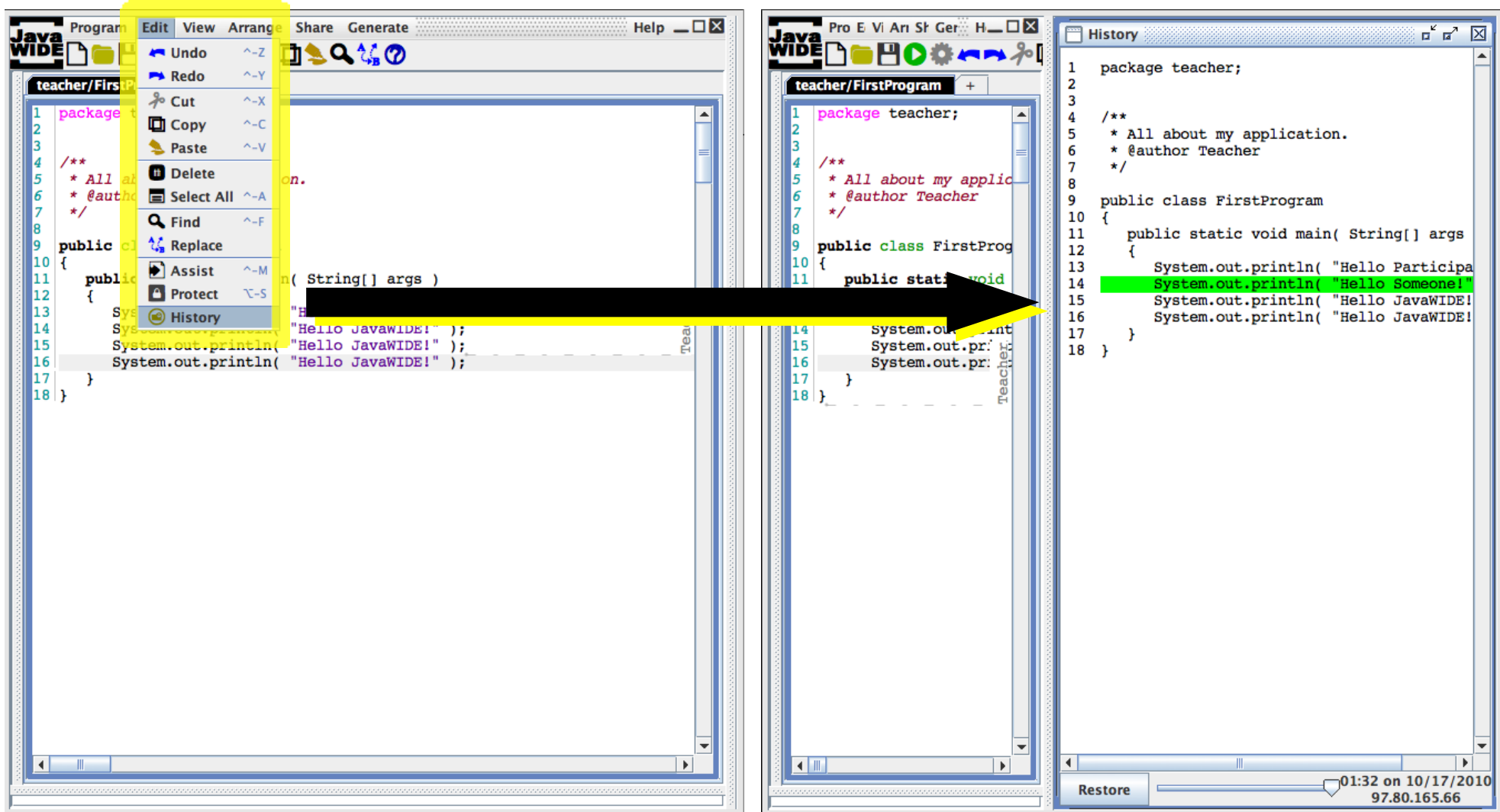
```
1 package teacher;
2
3
4 /**
5  * All about my application.
6  * @author Teacher
7  */
8 public class FirstProgram
9 {
10     public static void main(String[] args)
11     {
12         System.out.println("Hello Participant!");
13         System.out.println("Hello JavaWIDE!");
14         System.out.println("Hello JavaWIDE!");
15         System.out.println("Hello JavaWIDE!");
16     }
17 }
```

# History Demo

- *Motivations*
    - Revert to older version
    - Review program development
  - *Watch Demo*
- ## Desktop Sharing



# Open the History



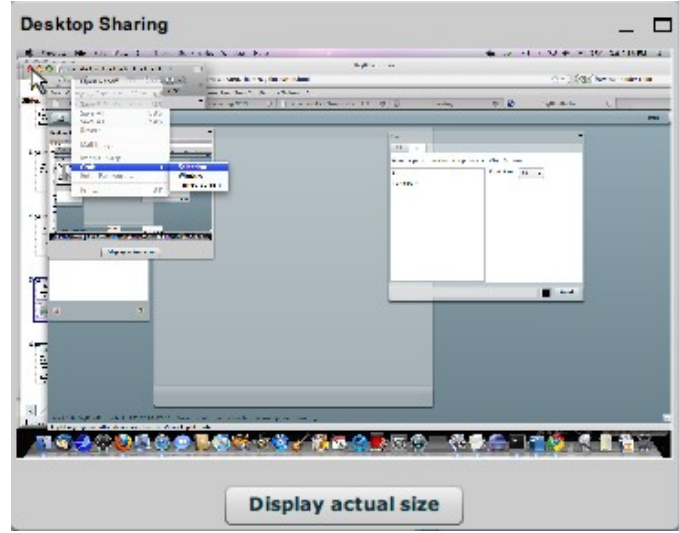
# Create Program Demo

- *Motivation*

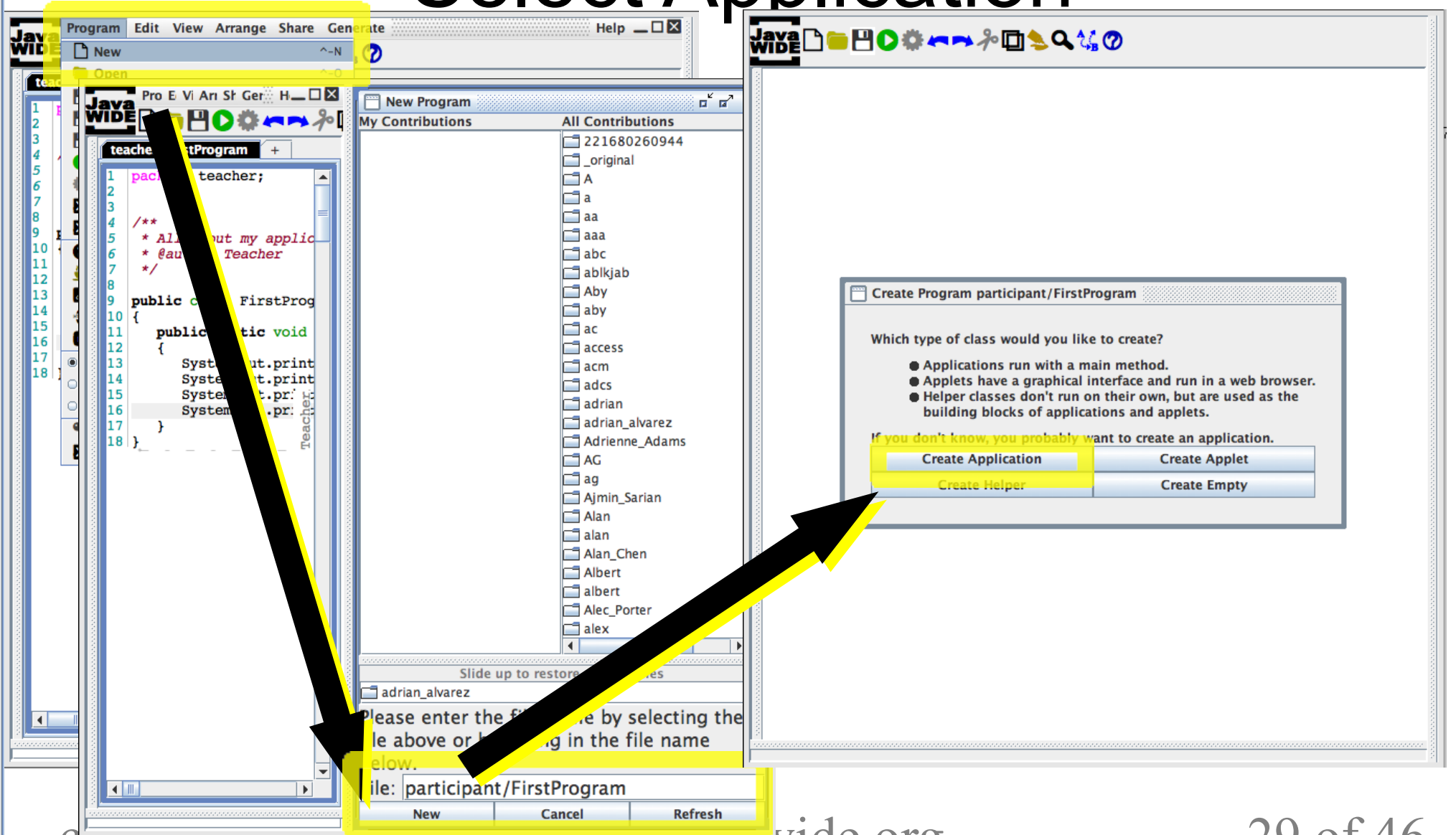
Make your own program

- *Watch Demo*

Desktop Sharing

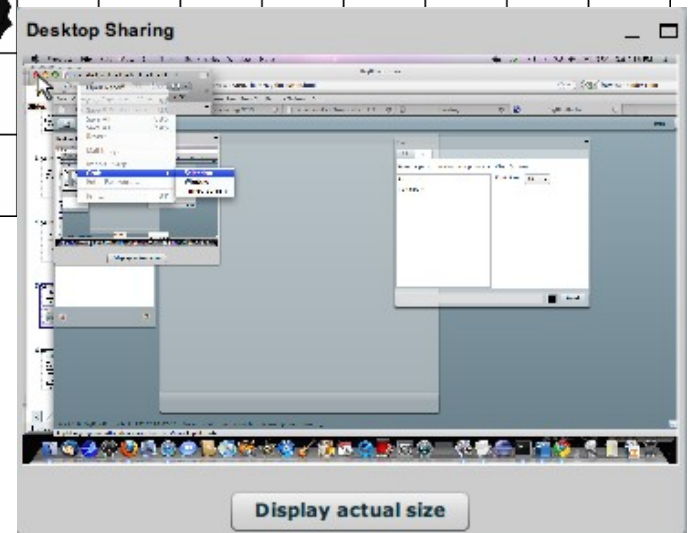
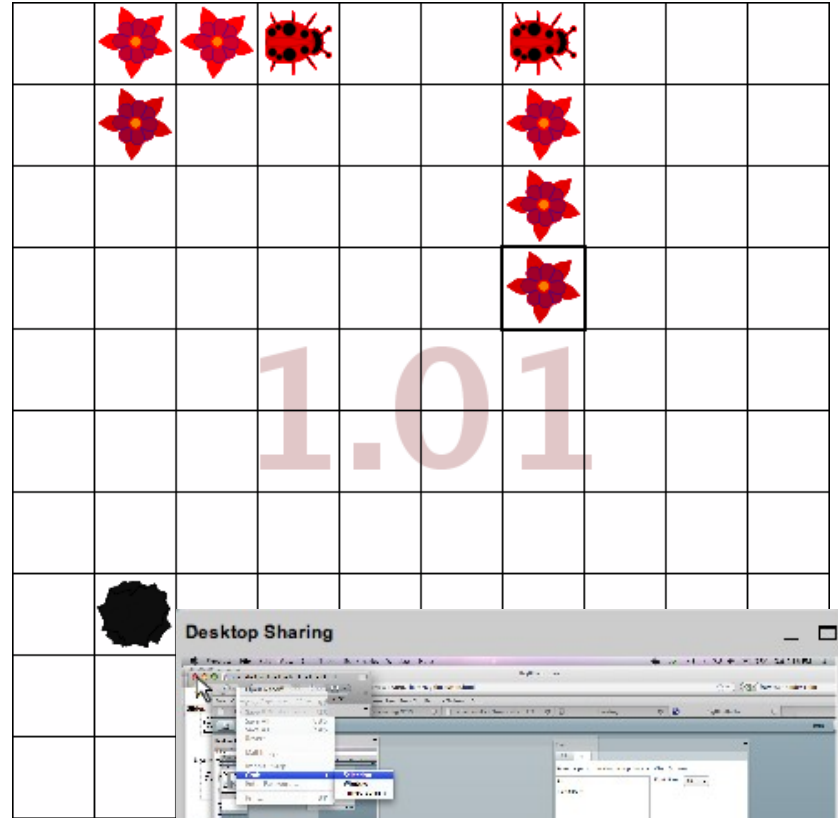


# Program → New, Name Program, Select Application

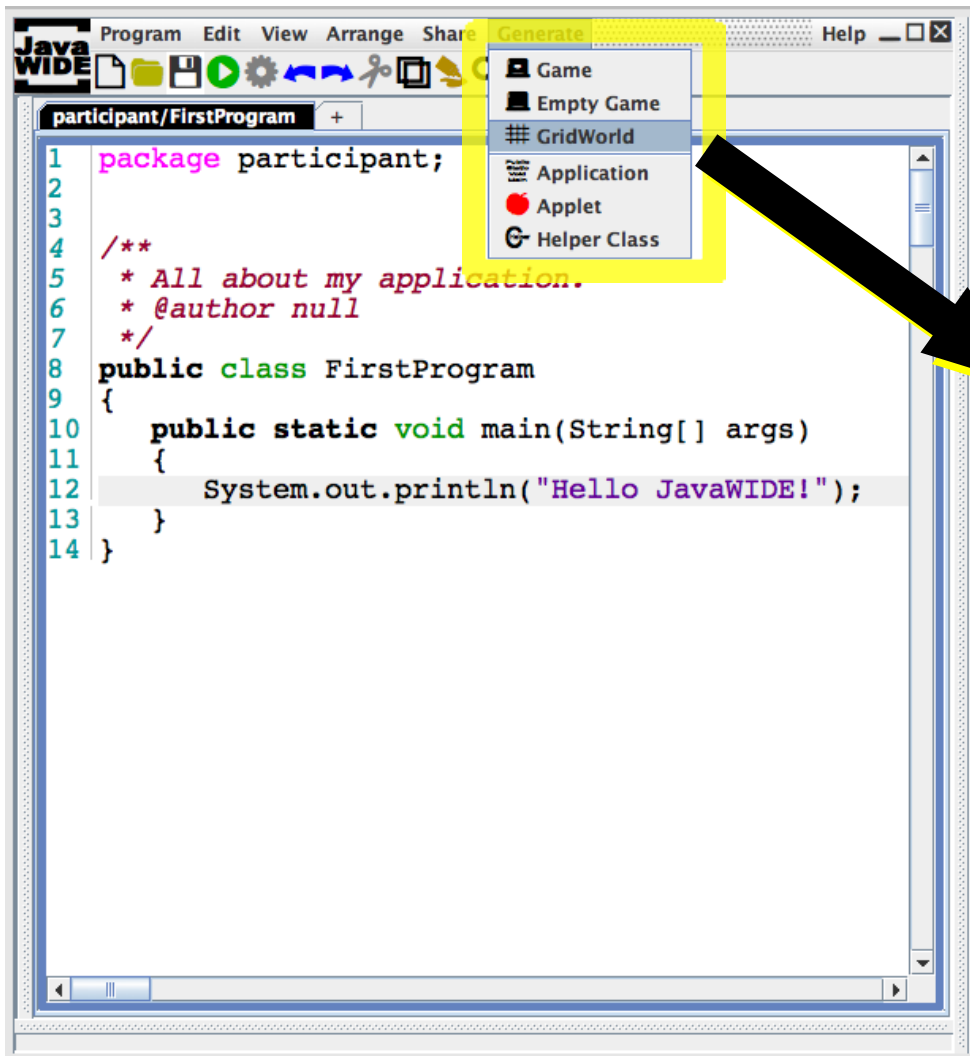


# GridWorld Demo

- *Motivation*  
AP Case Study
- *Watch Demo*  
Desktop Sharing



# Generate GridWorld Program



```
1 package participant;
2
3
4 /**
5  * All about my application.
6  * @author null
7  */
8 public class FirstProgram
9 {
10     public static void main(String[] args)
11     {
12         System.out.println("Hello JavaWIDE!");
13     }
14 }
```

```
1 package participant;
2
3
4 import info.gridworld.actor.*;
5
6 /**
7  * All about my application.
8  * @author null
9  */
10 public class FirstProgram
11 {
12     public static void main(String[] args)
13     {
14         System.out.println("Loading Grid...");
15         ActorWorld world=new ActorWorld();
16         world.add(new Bug());
17         world.add(new Rock());
18         world.show();
19         System.out.println("Loaded Grid.");
20     }
21 }
```

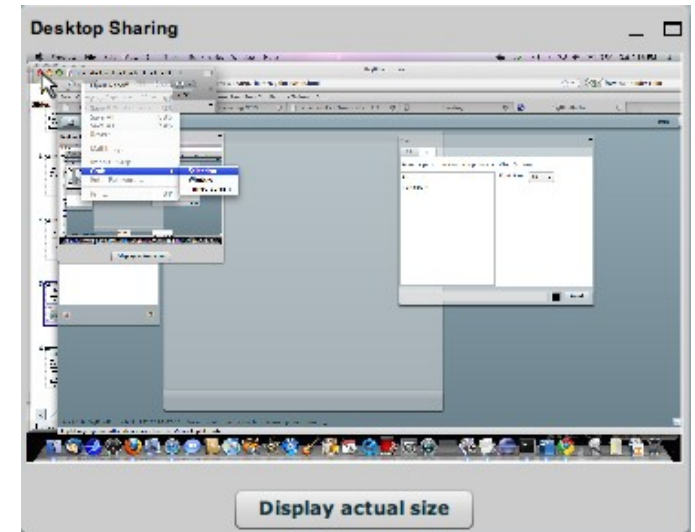
# Code Completion Demo

- *Motivation*

Why memorize fields & methods?

- *Watch Demo*

Desktop Sharing



# Press TAB After . (dot)

System.out.

[TAB]

[TAB]

```
System.out.print(Object)  
System.out.print(String)  
print(boolean)  
print(char)  
print(char[])  
print(double)  
print(float)  
print(int)
```

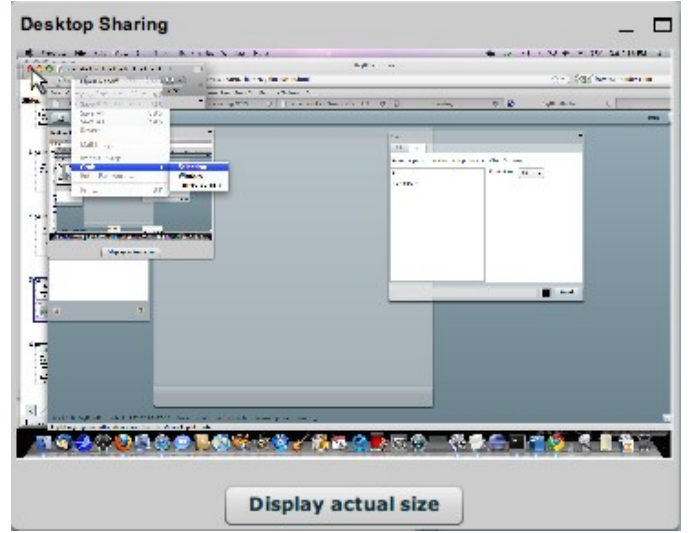
# Auto-Import Demo

- *Motivation*

Why memorize packages?

- *Watch Demo*

Desktop Sharing



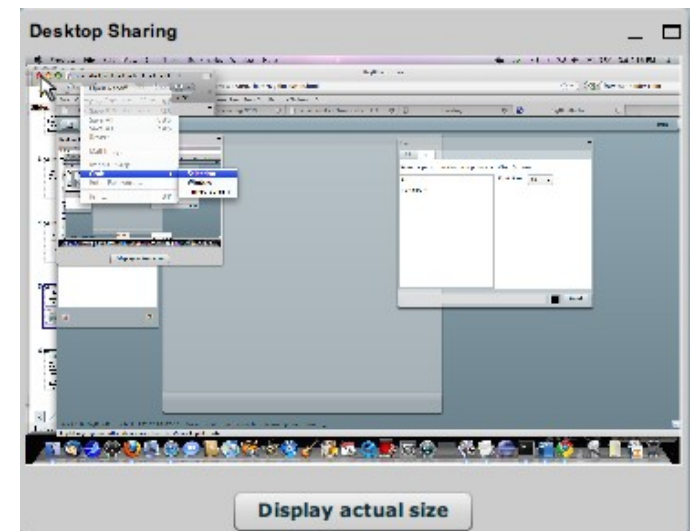
# Use ArrayList, import java.util

```
Java WIDE Program Edit View Arrange Share Generate Help
participant/FirstProgram +
1 package participant;
2
3
4 /**
5  * All about my application.
6  * @author null
7  */
8 public class FirstProgram
9 {
10     public static void main( String[] args)
11     {
12         ArrayList x
13         System.out.println("Hello JavaWIDE!");
14     }
15 }
```

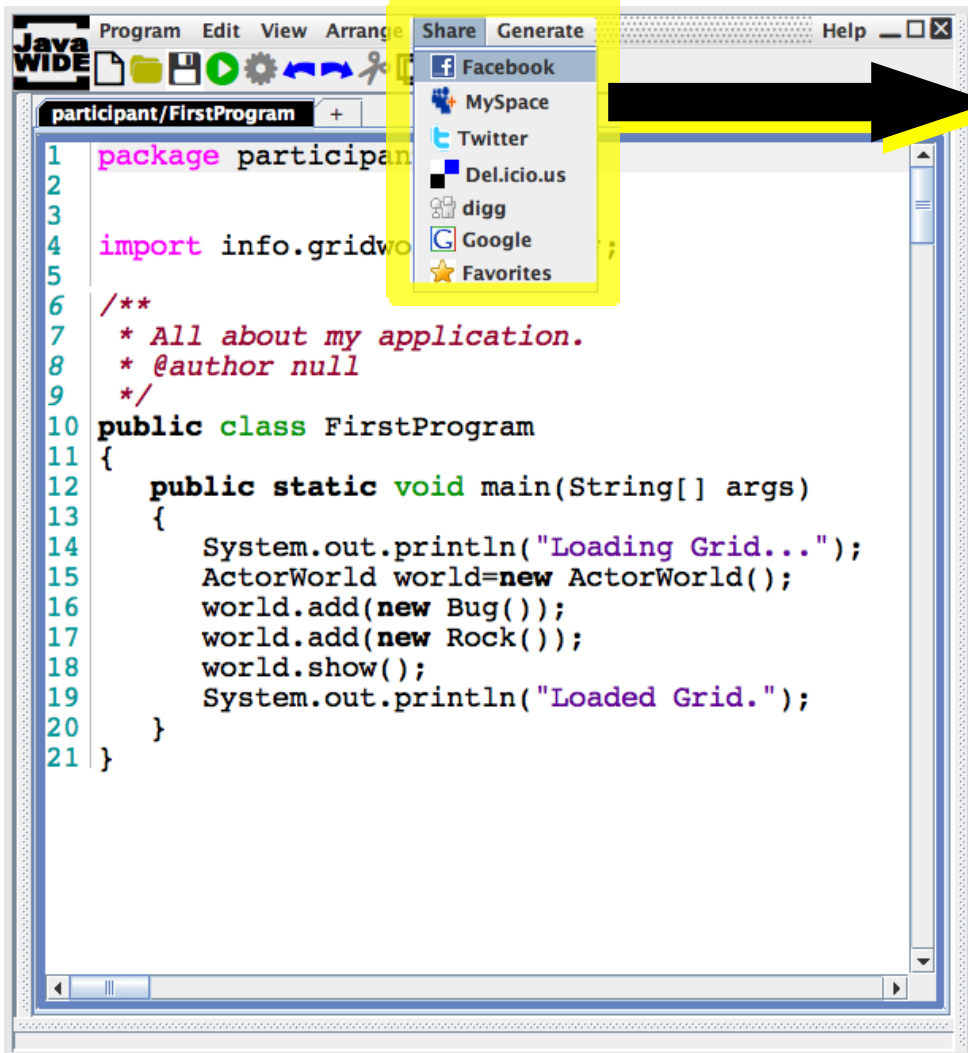
```
Java WIDE Program Edit View Arrange Share Generate Help
participant/FirstProgram +
1 package participant;
2 //start auto-imports
3 import java.util.*;
4 //end auto-imports
5
6
7 /**
8  * All about my application.
9  * @author null
10 */
11
12 public class FirstProgram
13 {
14     public static void main( String[] args )
15     {
16         ArrayList x;
17         System.out.println( "Hello JavaWIDE!" );
18     }
19 }
```

# Social Networking Demo

- *Motivation*
    - Show friends & family
    - Integrate with interest
  - *Watch Demo*
- ## Desktop Sharing



# Share with Facebook



```
1 package participant
2
3
4 import info.gridworld
5
6 /**
7  * All about my application.
8  * @author null
9  */
10 public class FirstProgram
11 {
12     public static void main(String[] args)
13     {
14         System.out.println("Loading Grid...");
15         ActorWorld world=new ActorWorld();
16         world.add(new Bug());
17         world.add(new Rock());
18         world.show();
19         System.out.println("Loaded Grid.");
20     }
21 }
```

Facebook Login

You must log in to share "participant/FirstProgram" with your friends.

Email:

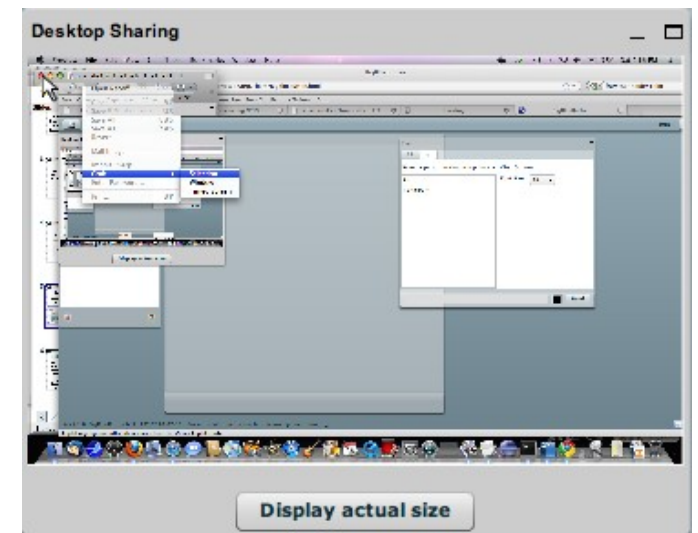
Password:

Keep me logged in

Sign up for Facebook

# Show Code Link Demo

- *Motivation*
    - Automatic program posting
    - Turn in assignment
  - *Watch Demo*
- ## Desktop Sharing



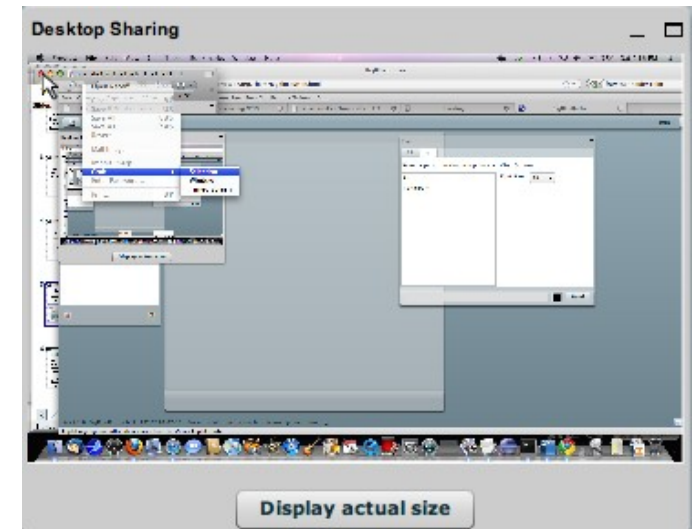
# Get the Web Address

The image shows a sequence of three windows from the JavaWide IDE. The first window is the IDE's menu, with the 'Show Code Link' option highlighted in yellow. A large black arrow points from this menu item to the 'Show in Browser' button in the second window. The second window shows the source code of a Java program. The third window is a browser displaying the URL <http://sandbox.javawide.org/index.php/participant/FirstProgram>.

```
1 package par
2
3
4 import info
5
6 /**
7  * All abou
8  * @author
9  */
10 public clas
11
12     public s
13     {
14         Syste
15         Actor
16         world
17         world
18         Syste
19     }
20
21 }
```

# Tooltip/Tutorial Demo

- *Motivation*
    - Explore program parts
    - Self guided learning
  - *Watch Demo*
- ## Desktop Sharing



# Read Tooltips & Tutorials

The screenshot shows the JavaWIDE website interface. On the left is a navigation menu with sections like 'Main Menu' and 'Toolbox'. The main content area displays a code editor for a file named 'FirstProgram.java'. The code includes package, import, and class declarations. A yellow box highlights the word 'public' in the class declaration 'public class FirstProgram', and a tooltip points to it with the text: 'public is used to indicate unrestricted access to the class members'. Below the code editor, there is a link to 'Download/View participant/FirstProgram.java'.

```
01 package participant;
02
03
04 import info.gridworld.actor.*;
05
06 /**
07  * All about my application.
08  * @author null
09  */
10 public class FirstProgram
11 {
12     public static void main( String[] args
13     {
14         System.out.println( "Loading Grid..." );
15         ActorWorld world = new ActorWorld();
16         world.add( new Bug() );
17         world.add( new Rock() );
18         world.show();
19         System.out.println( "Loaded Grid." );
20     }
21 }
22 }
```

The screenshot shows a web browser window displaying 'The Java™ Tutorials' page. The browser's address bar shows the URL 'http://download.oracle.com/jav...'. The page title is 'Controlling Access to Members of a Class'. The left sidebar contains a table of contents with links to various tutorial topics. The main content area features the title 'Controlling Access to Members of a Class' in red, followed by a paragraph explaining access level modifiers and a bulleted list of examples.

**The Java™ Tutorials**

Classes and Objects  
Classes  
Declaring Classes  
Declaring Member Variables  
Defining Methods  
Providing Constructors for Your Classes  
Passing Information to a Method or a Constructor

Objects  
Creating Objects  
Using Objects

More on Classes  
Returning a Value from a Method  
Using the this Keyword  
**Controlling Access to Members of a Class**  
Understanding Instance and Class Members  
Initializing Fields  
Summary of Creating and Using Classes and Objects  
Questions and Exercises  
Nested Classes  
Inner Class Example  
Summary of Nested Classes  
Questions and Exercises  
Enum Types  
Questions and Exercises

Home Page > Learning the Java Language > Classes and Objects  
« Previous • Trail • Next »

## Controlling Access to Members of a Class

Access level modifiers determine whether other classes can use a particular field or invoke a particular method. There are two levels of access control:

- At the top level—public, or package-private (no explicit modifier).
- At the member level—public, private, protected, or package-private (no explicit modifier).

A class may be declared with the modifier public, in which case that class is visible to all classes everywhere. If a class has no modifier (the default, also known as package-private), it is visible only within its own package (packages are named groups of related classes—you will learn about them in a later lesson.)

At the member level, you can also use

# Stay Organized

- Package is lowercase username
- Store programs in own package
- Organize example classes within packages with date:  
`teacher/oct19/FunExample`  
`teacher/oct19/Example5`

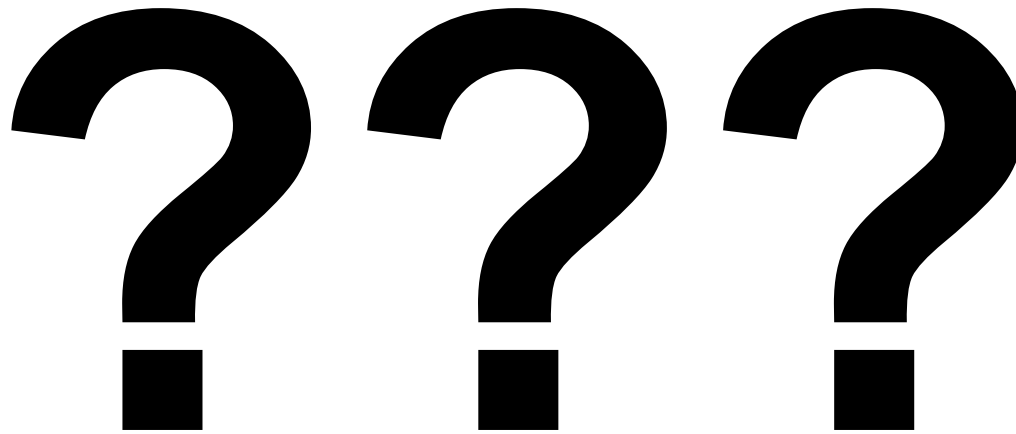
# Resources

- JavaWIDE Forum:  
<http://forum.javawide.org>
- How-to Videos:  
<http://csc.javawide.org>
- Developers' Blog:  
<http://developers.blog.javawide.org>

# Conclusion

- Free, online, easy-to-use IDE
- **Zero installation, zero maintenance**
- Innovative & unique features
- Promotes collaboration & engaging classroom examples
- Available now for high schools

# Discussion & Questions



**Digital Ink Demo**

# Next Steps

- Practice in JavaWIDE Sandbox
- Create interactive examples with your class in the Sandbox
- Get your own JavaWIDE site
- Attend an upcoming webinar
- Request another webinar