Distributed Collaboration - Assignment 3: N-User Shared Windows with Customizable Telepointers

Date Assigned: Sunday September 28, 2014

Submission Date: Wednesday Oct 8, 2014(11:55 pm)

Objectives:

- Understand how to implement a general infrastructure/abstraction
- Understand implementation of a telepointer and graphics
- Understand implementation of near-WYSIWIS coupling
- Understand how to automatically identify corresponding replicas

So far, you have created application-specific designs and implementations. In this project, you will learn how to implement a general collaboration infrastructure/abstraction on top of an existing infrastructure/abstraction. Specifically, you will build a shared window system with customizable telepointers on top of the Java AWT/Swing layer, which will implement near-WYSIWIS coupling. You should use my embellishment of AWT/Swing though you are free to modify the library code or implement directly on top of Swing/AWT. As before you will trace certain predefined steps. As extra credit, you can allow mouse drag events to be filtered out depending on a customization parameter that can be changed interactively.

You will need to pull the latest changes from ColabTeaching for this project. Before each assignment, be sure to pull any changes that have been made.

All assignments will most probably be changed after they are posted to clarify them or change constraints, and features. The changes will be highlighted using track changes.

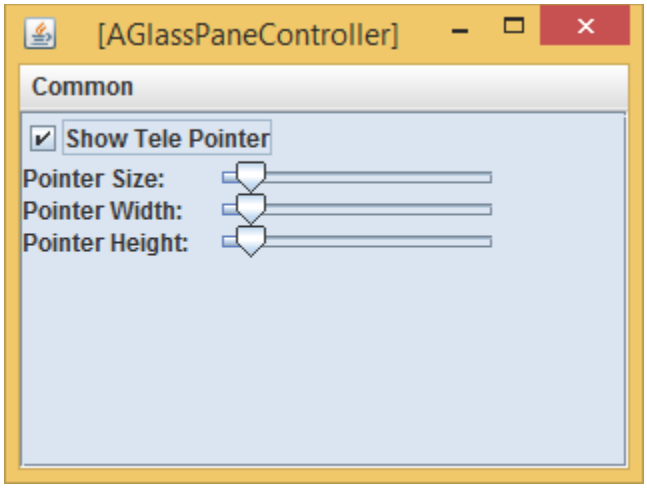
You will demonstrate your solution by sharing the window(s) created by the single-user IM tool – not your multi-user IM tool as that provides its own sharing. If you no longer have your assignment 1, you get a copy of the version you submitted to me.

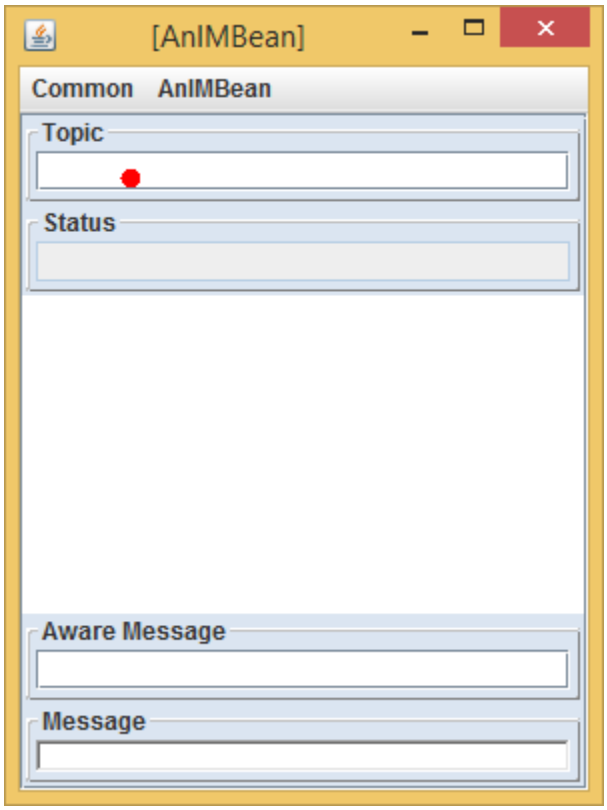
Customizable User-defined Frame Pointer

Create a pointer that can be moved anywhere on (the Swing components of a) Swing JFrame. If you use my library, then this means you need to simply create an object that paints a shape of your choice. In my demo I drew a red circle – you are free to draw any shape and color it anyway

you like. This shape will, of course, become a telepointer when you implement the next part of the assignment.

Your painter should use the relevant properties of the library ExtendibleTelepointerGlassPane, Width, Height and ShowTelePointer in its paint method. Create a separate user-interface for displaying and setting these three properties. The pointer-customization user-interface should be displayed in a frame that is separate from the frame being pointed. You can assume that only one frame has a pointer. The YouTube demo shows my implementation of the pointer-customization frame, reproduced below along with the pointed frame. You can ignore the pointer size control – in my implementation it is used to control the pointer when it is circular.





Multi-User Replicated Shared Window System

Now allow a pointed frame to be shared by multiple users. Implement the replicated architecture. Your system should provide the application programmer an API to exclude certain frames from being shared based on their names. Test the API using your customization frame – confirm that you can turn on and off the sharing of the frame. In your demo, show both frames being shared. You will use private windows in the next assignment in which you will not share the locking frame.

Your implementation will need to find corresponding replicated windows. You can determine this correspondence based either on window creation sequences or window names. In the latter case, you will need to name each component of each window tree. In both cases, you will need to register each component, that is, associate the local component object with a global id.

Extra Credit: Mouse-Drag Filtering

Provide an API to filter out mouse-drag events about based on differences in the times when these events are generated. Define some threshold T and do not send a mouse drag to a remote computer if the last drag was sent less than T time units ago, while ensuring that the last mouse drag event is sent (before the next non- drag event). Allow time T to be customized using a user-interface that can be displayed as part of the pointer-customization frame. If you integrate the two user-interfaces, you are free to determine whether the threshold applies to all shared frames or only the frame associated with the customized pointer. If you create a separate filtering frame, you are free to determine if it is a private or shared window in the demo, and which frames the threshold applies to.

Tracing

You should now fire three traceable events: ComponentTreeRegistered, AWTEventSent, and ReceivedAWTEventDispatched. ComponentTreeRegistered is fired after you register all components of a window subtree and should pass the triggering resize event (for the root of the subtree) to the newCase() method. AWTEventSent is fired just before the multicast of a local event is initiated. ReceivedAWTEventDispatched is fired just before a remote event is delivered to the local input queue. You are free to but not required to fire AWTEventReceived.

Make sure the tracing in on in your clients for the events you fire. To do so, you should execute the following code before you display your user interface:

```
Tracer.showInfo(true);
SharedWindowTracerSetter.traceSharedWindow();();
```

Main Class and Tags

As in the last assignment you will provide and run at least N+ 1 main classes, one for the session manager, and one for each of the N clients in a session. As in the code and demo shown in lectures, you should support at least 3 clients in the session. As in the lecture code, the session manager main class should have the tags: *DistributedTags.SESSION_MANAGER, ApplicationTags.REPLICATED_WINDOW*, and the main class for client^{k} should have the tags: DistributedTags.*CLIENT_k, ApplicationTags.REPLICATED_WINDOW*. Both kinds of classes can have additional tags such as *DistributedTags.SERVER* for the session manager.

Submission

By the submission date, submit a link to a video (with an audio narration) showing your shared window system in action and also submit your code to Sakai. Please post the link as a private message in Piazza rather than an email (which is hard to keep track of) or a Sakai submission (the overhead of logging in to Sakai is high). Tag the link as hw3videolink. If you are worried about privacy issues, free to use the other means, and post a Piazza message informing me you have done so