Connor Jackson, Adam Claxton

Assignment #3 CSE2102

Rapla Scheduler – Fast Resource Search

October 11, 2015

1. In this project, we will be adding a fast search feature to the resources section of this application. We are interpreting this as picking a type of resource, choosing an attribute to search by, and typing in what we want to search. Then, in the calendar view, only events containing that resource-attribute match will be displayed. This is very similar to the filter feature, except you have to uncheck every single resource you don’t want to see if you only want to view one. Our addition will make displaying a single resource much easier.  
     
   Since we can use a similar implementation as the filter feature, we could use inheritance from FilterEditButton located in org.rapla.gui.internal. This creates a small button from the RaplaArrowButton class that looks like such:



We would have to change the text to read “search” though.   
  
Along with this, we would like to use the class ClassifibleFilterEdit located in org.rapla.gui.internal.edit. This class seems to help create text fields and combo boxes inside the popup a filter button would have, which helps in our search feature. We would like our user to choose a resource to search through a combo box, then an attribute through a combo box, then type in their criteria through a text field.  
  
We are also thinking about using ClassificationFilterRuleImpl & ClassificationFilterImpl from org.rapla.dynamictype.internal or ClassificationFilterRule & ClassificationFilter from org.rapla.dynamictype. These parts of the program seem to contain the code that uses filter rules that choose which events in the Rapla interface fit defined rules. In this case, we can manipulate it to match events with search criteria.  
  
Lastly, we will need to change the code a bit in the section of this project that actually adds each part of the program to the interface frame. This is simply where we will add the search arrow button.

1. The existing code helped a lot in our design process. Our first gut reaction to working on this assignment at first was to explore the program and see what features already existed, from a user perspective. Once we found how closely resembling the filter feature was to search, we knew we should change that around to make our work progress well. This way, we won’t have to develop any algorithms for searching and just use ones that already work perfectly fit with the program.  
     
   Also, the project was designed in such a way that most of the code is generalized and can be reused in many different cases. This makes it very easy for extra add-ons to be implemented, as it seems designed to be easy for others to collaborate on.