Software Difference Analyzation Tool

Rolf Biehn

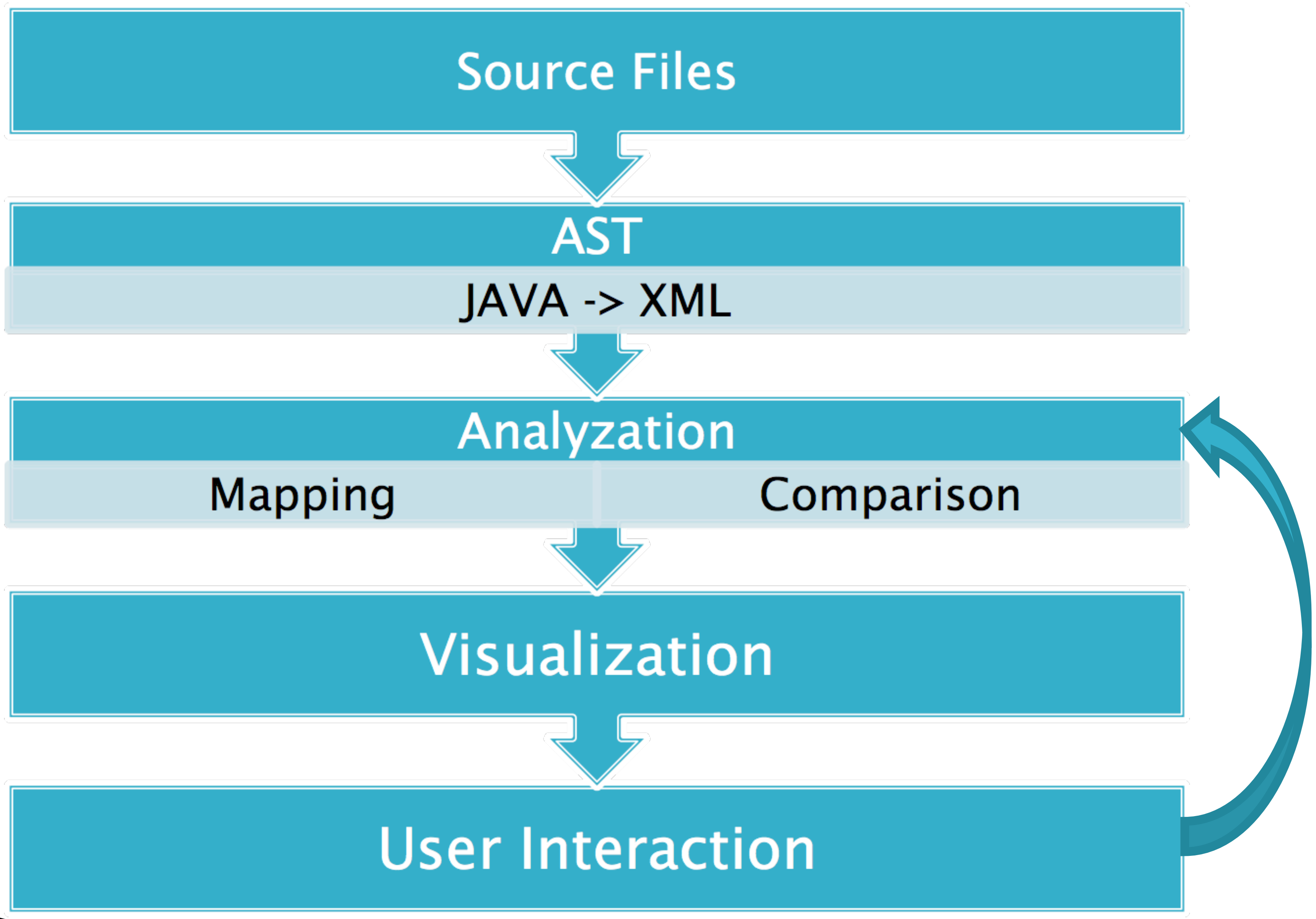
The Problem

- Investigation of a recently introduce bug
- Code review code before a check-in
- Clone Detection

Lots of file-to-file text solutions to this problem exist, but we can improve it…

My Solution

- Use an Abstract Syntax Tree(AST) and structurally compare code
  - Reduce noise (such as renamed variables, method declaration order, etc..)
  - Cross-language comparisons become possible
  - Better visualizations possible
  - Improved Navigation
- Complete solution is out of scope


Source Files
AST
JAVA -> XML
Analyzation
Mapping
Comparison
Visualization
User Interaction

AST Generation/Analyzation

- Data set is source code found from the web
- Using JAVA2XML for AST conversion
  - Some issues & limitations discovered
- Using my own comparison algorithm

Example (Source)

```
public static void print() {
		System.out.println("Hello World!");
}
```

```
public static void print() {
		System.out.println("Hello Universe!");
}
```

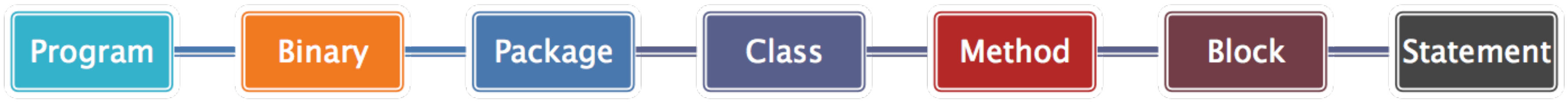
Example (After AST)

```
<method name="print" visibility="public" static="true">
   <type name="void" primitive="true"/>
   <formal-arguments/>
      <block>
         <functionCall="println">
            <target>
               <field-access field="out">
                  <var-ref name="System"/>
               </field-access>
            </target>
            <arguments>
              <literal-string value=""Hello World!""/>
            </arguments>
        </functionCall>
     </block>
</method>
```

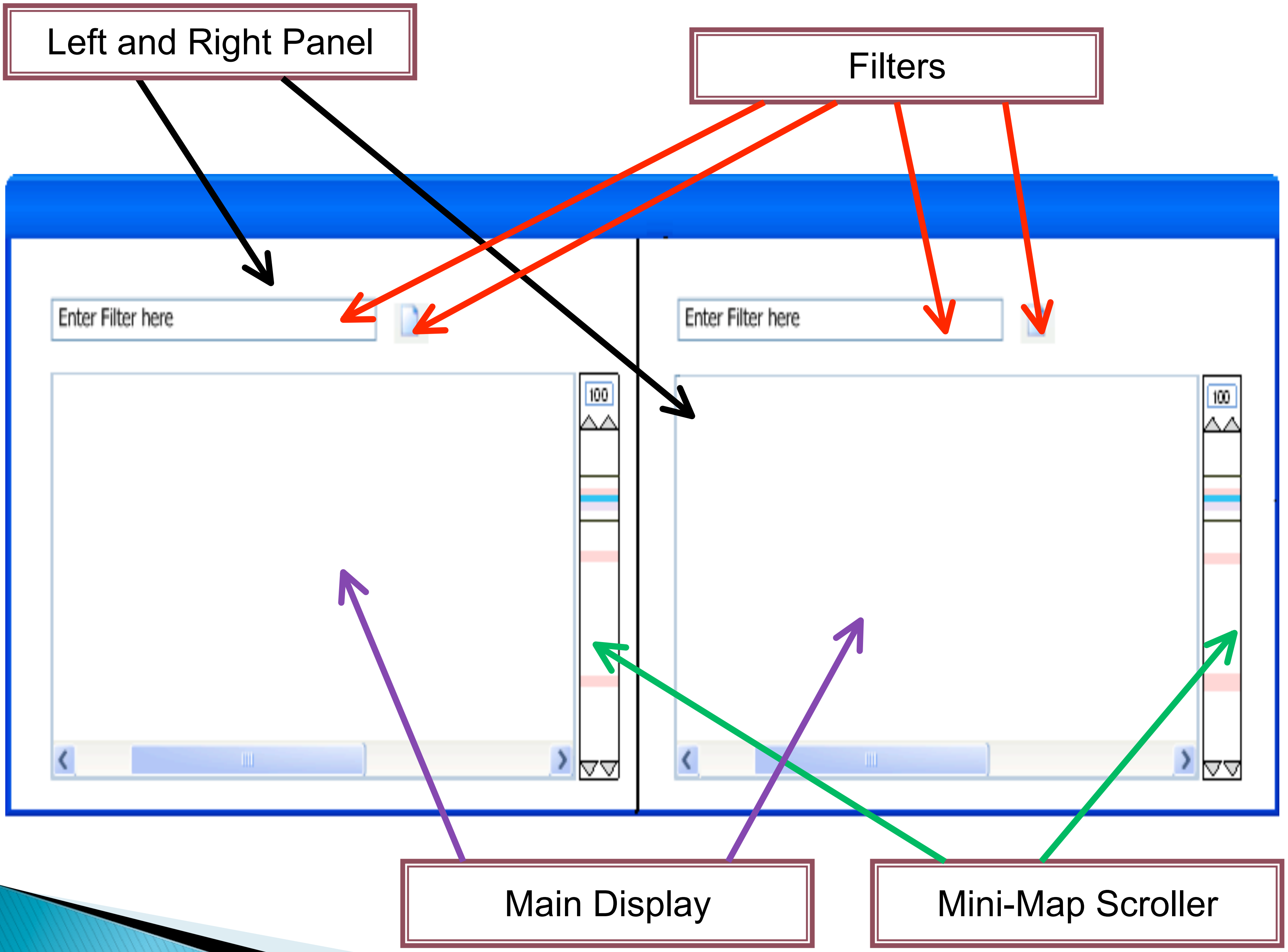
Example (After Analysis)

```
<method name="print" visibility="public" static="true">
   <type name="void" primitive="true"/>
   <formal-arguments/>
      <block>
         <functionCall="println">
            <target>
               <field-access field="out">
                  <var-ref name="System"/>
               </field-access>
            </target>
            <arguments>
              <literal-string value=""Hello World!“”>
                 <metaInfo type=“diff”/>
              </literal-string value>
            </arguments>
        </functionCall>
     </block>
</method>
```

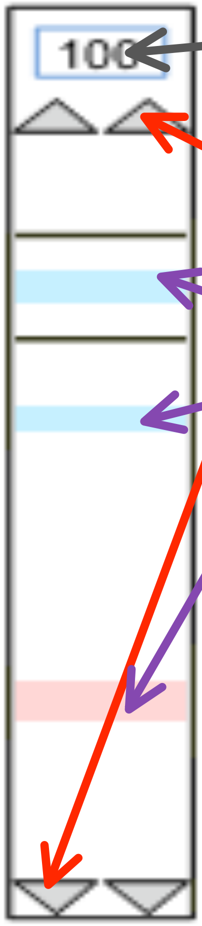
Visualization



- The Idea is multiple (specialized) views at each level of the (simplified) hierarchy
- Focus for this assignment is on Method and below


Left and Right Panel
Filters
Enter Filter here
Enter Filter here
100
100
Main Display
Mini-Map Scroller

Mini-Map Scroller



Users click here to change the percentage zoom factor.

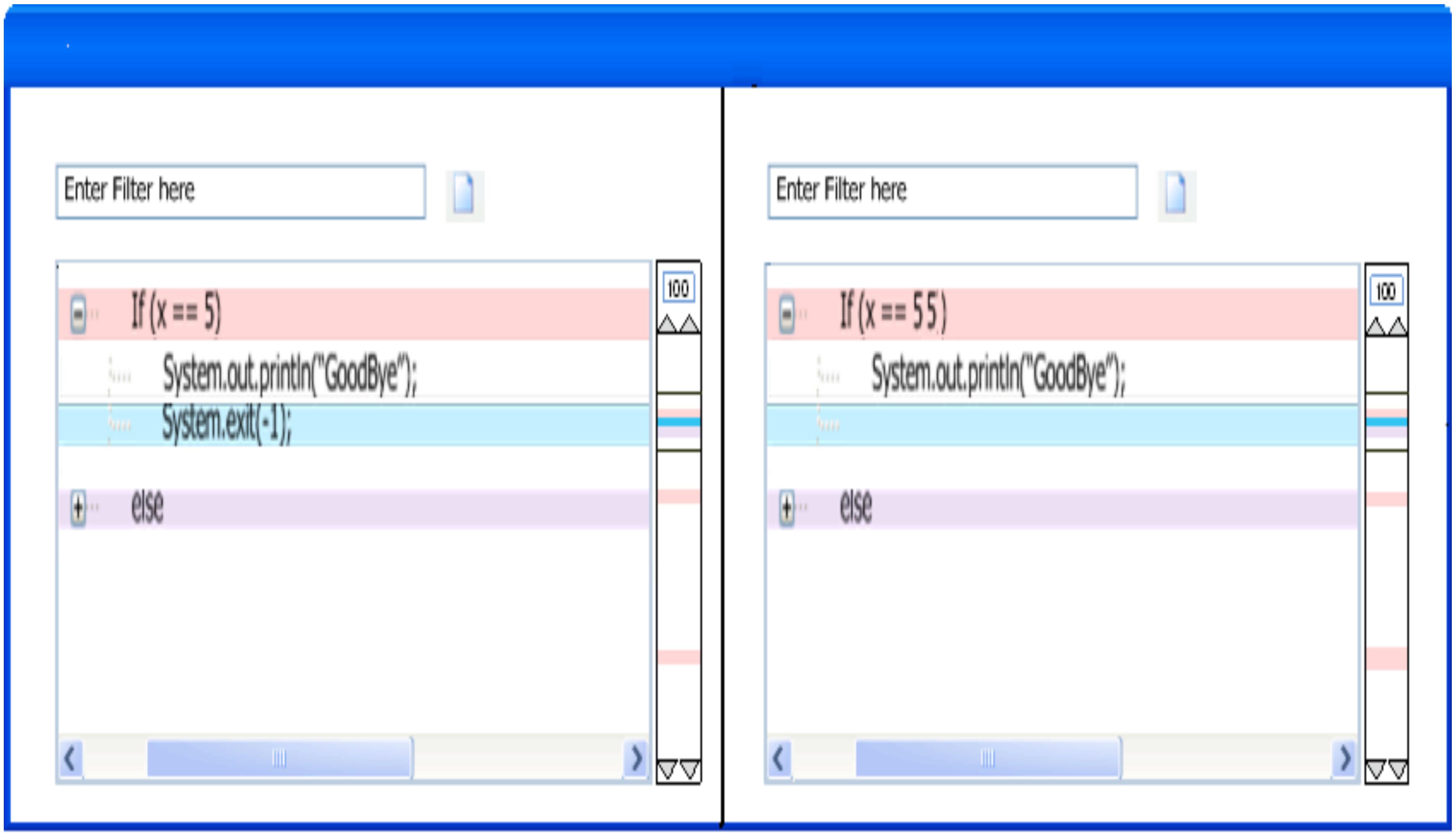
Arrows for Page Up and Page Down.

Lines used to indicate interesting areas in the main display (i.e. red for diff, blue for orphans), current display region also shown. % of width used if # of lines is insufficient.

Scroller holds mouse until another click or esc.

Moving mouse orthogonally changes zoom factor, moving mouse 3 moves to the next difference.

Method Comparison



Enter Filter here

```
If (x == 5)
    System.out.println("GoodBye");
    System.exit(-1);
else
```

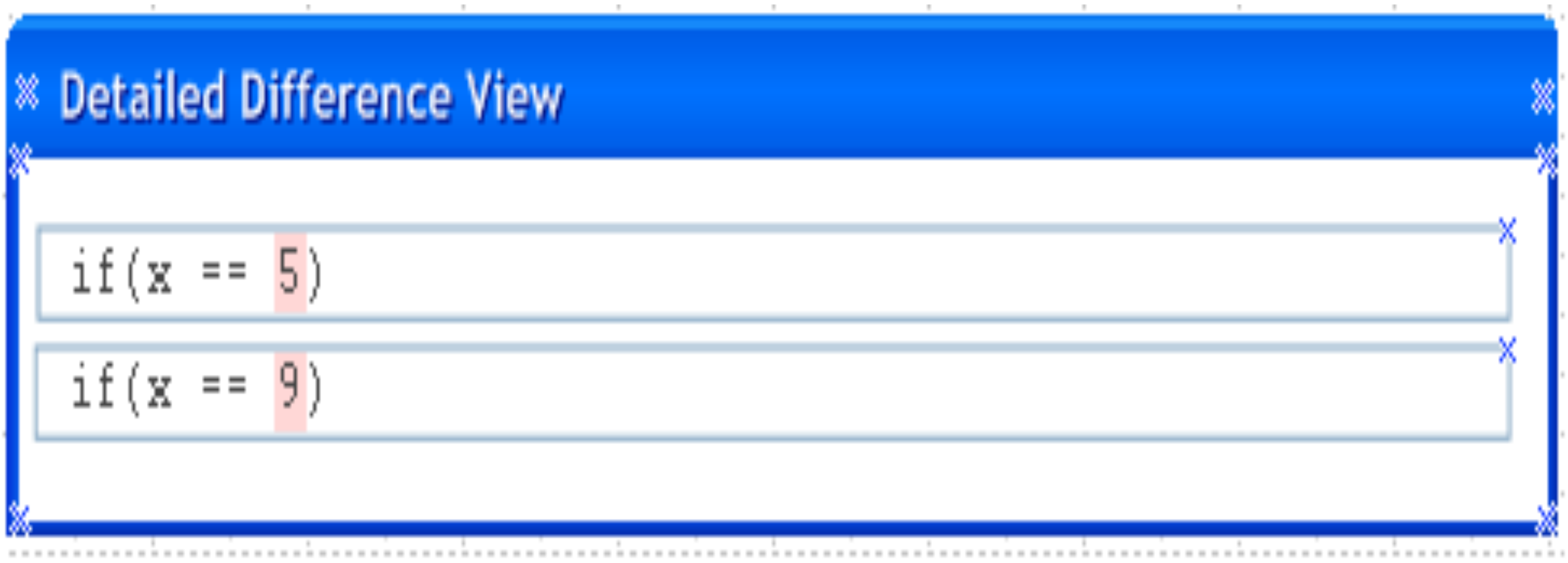
Enter Filter here

```
If (x == 55)
    System.out.println("GoodBye");

else
```

Red=diff, blue=orphan, purple=hidden child diff

Detailed Difference View



```
if(x == 5)
```

```
if(x == 9)
```

Progress

- AST and Comparison Stage Complete
- Mini-Map Scroller progress slow
- Overall a little behind, but I am adjusting

Questions?