

# Eclipse Language Plugin

---

Michael Ferdinand Moser (1123077)

Frederic Golser (1230316)

Moritz Stumpfegger (1223517)

# Project Task

---



- Create an Eclipse CDT PlugIn that enables the user to hide certain lines of code using regular expressions.
- Furthermore, the PlugIn is to support highlighting of source code.
- Finally, we were to research ways to modify the debugger such that it skips lines of source code that have previously been hidden.
- Input language: C

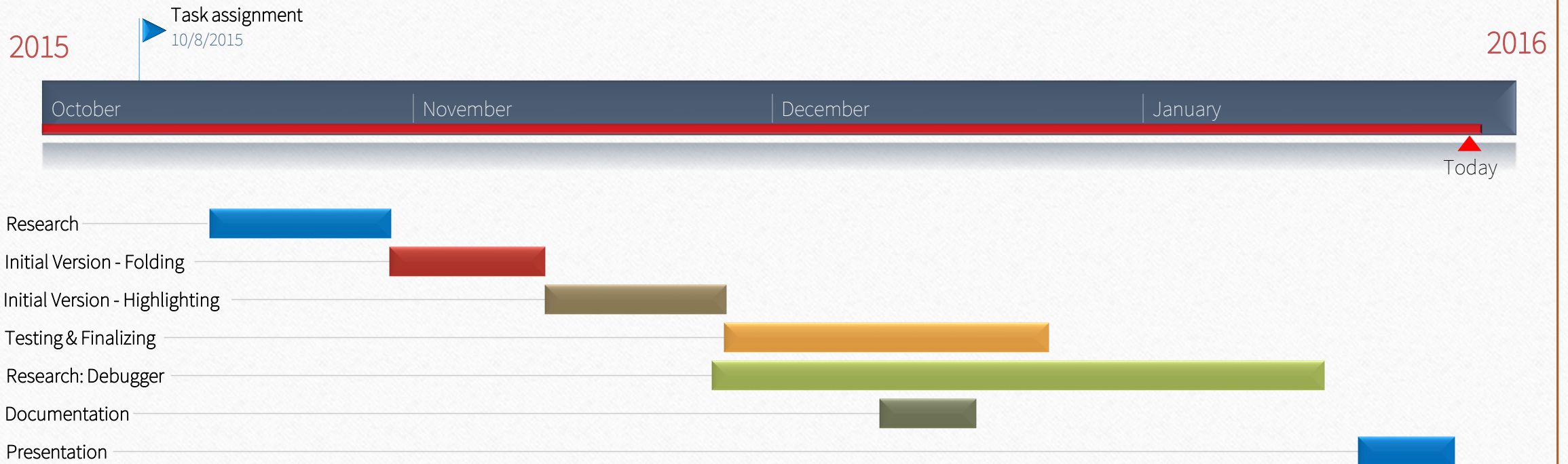


# WorkFlow



- 
- We used GitHub as preferred VCS.
  - We had meetings approximately every two to three weeks to discuss current issues.

# WorkFlow



# Eclipse PlugIn Development

---



- Eclipse itself can be considered to be a base workplace which loads various plugIns.
- You can get started by creating an empty plugIn development project within Eclipse.
- PlugIns are *self-describing* (`MANIFEST.MF`) and *self-hosting* (`plugin.xml`).



# Eclipse PlugIn Development



- *Self-describing:* The MANIFEST.MF file contains information about the plugin, as well as dependencies and packages offered to other plugins.

```
Manifest-Version: 1.0
Bundle-Name: C/C++ Regex Folding Plugin
Bundle-SymbolicName: <plugin-name>;singleton:=true
Bundle-Activator: <fully qualified name>
Require-Bundle: org.eclipse.ui,
org.eclipse.cdt.debug.ui;bundle-version="7.5.0"
org.eclipse.ui.editors;bundle-version="3.8.200"
Bundle-RequiredExecutionEnvironment: JavaSE-1.8
Bundle-ActivationPolicy: lazy
Bundle-Vendor: <our names>
Export-Package: <list of packages>
```

# Eclipse PlugIn Development



- *Self-hosting*: plugin.xml provides a list of **extension points**.
- Run the plugin.xml file as an Eclipse Application to test your plugin.

```
<plugin>
...
<extension point = "org.eclipse.ui.editors">
  <editor
    class=<fully qualified name>
    id=<unique name>
    name="Regex C/C++ Editor">
  </editor>
</extension>
...
</plugin>
```



# Folding/Highlighting Algorithm

---



- Create a map  $M$  for the content of the current editor such that  $M[i]$  contains the index of the first character in the  $i$ -th line.
- Feed the content of the editor to some `Matcher` object and retrieve the indexes of the matches.
- Use map  $M$  to look up in what lines of source code those matches occur.



# Folding Source Code

---



- Challenge: How do we get hold of the text of the current editor?
- Challenge: How are we supposed to implement the methods of the `ICFoldingStructureProvider` interface?
- Question: How to let user activate folding of code lines?
- Use `PreferenceStore` to save user-specified settings.

# Highlighting Source Code

---



- Challenge: Interference with syntax highlighting.
- Challenge: Switch from folding to highlighting.
- Idea: Let user decide foreground/background colours.



# Modifying the debugger



- Ideally the debugger can be modified such that certain lines of code are skipped.
- Unfortunately, no easy extension point in the `org.eclipse.cdt.debug` package to implement this behavior is available right now.
- Alternatives: Write debugger from scratch, copy&paste existing code, ...
- How does the debugger work conceptually?

# How the debugger works...



- The `GDBCDIDebugger2` creates an `MIProcess` for the running `gdb` executable.
- After creating the `MISession`, the `GDBCDIDebugger2` creates a `Session` object to manage the operation of the CDI. The `Session` creates many objects to assist with its management responsibilities, most importantly the target object.
- The `Target` receives debugging commands from the CDT and packages them.



# Operation Example Target



- Step Over button: The CDT finds the current Target and calls its `stepOver` method.
- Target creates a `MIExecNext` command and calls `MISession.postCommand()`.
- `MISession` adds the Command to `CommandQueue`.
- `TxThread` that sends commands from the output `CommandQueue` to the `OutputStream`.

# Conclusion: Debugger

---



- “CDT debugger is so complicated that you may wonder whether any of the developers were related to Rube Goldberg”  
~ *ibm.com* 24 Jun 2008



# References

---



<https://github.com/mfm92/RegexHider>

<http://www.eclipse.org/ecd/img/eclipse256.png>

<http://www.ibm.com/developerworks/library/os-eclipse-plugindev1/index.html>

<http://help.eclipse.org/juno/index.jsp?topic=%2Forg.eclipse.platform.doc.isv%2Freference%2Fextension-points%2Findex.html>

[https://de.wikipedia.org/wiki/Eclipse\\_%28IDE%29#Architektur](https://de.wikipedia.org/wiki/Eclipse_%28IDE%29#Architektur)



---

Any questions?