Scala+GWT

Grzegorz Kossakowski
Google Atlanta summer intern, University of Warsaw
Why Scala in GWT?

- Scala brings modern development techniques to Java developers so why shouldn’t GWT developers benefit from it?

Java

```java
// Lists
TreeItem catLists = mainMenu.addItem(
    constants.categoryLists());
setupMainMenuOption(catLists,
    new CwListBox(constants),
    images.catLists());
setupMainMenuOption(catLists,
    new CwSuggestBox(constants),
    images.catLists());
setupMainMenuOption(catLists,
    new CwTree(constants),
    images.catLists());
setupMainMenuOption(catLists,
    new CwMenuBar(constants),
    images.catLists());
// and one more call that didn’t fit
```

Scala

```scala
// Lists
val catLists =
    mainMenu.addItem(constants.categoryLists)
    List(new CwListBox(constants),
    new CwSuggestBox(constants),
    new CwTree(constants),
    new CwMenuBar(constants),
    new CwStackPanel(constants)) foreach {
    setupMainMenuOption(catLists, _, images.catLists)
}
```

profit!!

interoperability with Java
functional programming
improved OO
Two compilers to be connected

- Both compilers work with **internal** data structure called Abstract Syntax Tree
- Essentially, we need to translate Scala’s ASTs to GWT ASTs
- Direct translation makes both projects tightly coupled

AST is internal representation of programs

Tight coupling would force both projects to coordinate often, which is far from being desirable.

GWT has the same situation with JDT, but Java evolves a lot slower.
Loose connection

• We need to connect both compilers in looser manner
• We need stable API that both projects can target
• Another language can be such a stable API, we call this language *jribble*. 

*text formats seems to work very well as a way to exchange data*
But what about Java source code?
Scala to Java translation proved to be very hard

- If it was easy someone would have already implemented it (others tried it already)
- Scala has constructs that are hard (or maybe impossible) to translate to Java
- all exceptions are unchecked
- constructor calls doesn’t follow Java rules (think of mix-in composition)
But what about Java byte code?
 Byte-code as stable API

• Nested expression structure is lost
  • t1 = bar
  • t2 = baz
  • t3 = t2 + 1
  • t4 = foo(t1,t3)
  • x = t4

• Arbitrary control flow is allowed
  • goto is harmful

x = foo(bar, baz+1)
Jribble

• Java-derived
• Jribble removes
  • package declarations and imports
  • default constructors
  • for loops
  • restrictions on calling super constructors
• Jribble adds
  • method/fields signatures for references

It's like a puddle of Java. It's also like drivel, which is appropriate enough for a language no one writes and no one reads. Computers can entertain themselves with it well enough.
– Lex Spoon

Jribble files are verbose enough to be parsed separately.
Scala, Jribble and GWT

Scala, Jribble and GWT

Scala, Jribble and GWT
public final class Lcom/google/gwt/sample/jribble/client/Hello$$anon$1;
    extends Ljava/lang/Object;
    implements Lcom/google/gwt/event/dom/client/ClickHandler; {

    public V; onClick(Lcom/google/gwt/event/dom/client/ClickEvent; event) {
        Lcom/google/gwt/user/client/Window;.
        (Lcom/google/gwt/user/client/Window;::alert(Ljava/lang/String;)V;)
        ("Hello, AJAX, said Scala");
    }

    public this(Lcom/google/gwt/sample/jribble/client/Hello; $outer) {
        (Ljava/lang/Object;::super()V;)();
    }
}
scalac side

• another backend (in addition to jvm, msil, etc.)
• isolated from the rest of the compiler
• includes both transformations and printing of trees
• extends scalac’s testing framework called parTest
• implemented in Scala
scalac challenges

• many things that are expressions in Scala are not in Jribble, e.g. block, if

• pattern matching logic emits trees that contain arbitrary jumps, including forward jumps

• Scala unifies types and we need to revert that. Nothing and Unit are problematic, e.g.:

  scala> true || (throw new RuntimeException)
  res0: Boolean = true
jribble side

• language specification (still to be formalized)
• library that defines AST nodes, parsers and printers
• written in Scala
• uses Scala’s Parser Combinators library including Packrat Parsers extension

```scala
def whileStatement: Parser[While] = opt(name <~ ":") ~
("while" ~> "(" ~> expression <~ ")") ~ block ^^ While
```
jribble side

• uses ScalaCheck framework for testing, Parser is just String => AST, and if we take Printer AST => String, we obtain equation:
  \[(\text{AST} \Rightarrow \text{String}) \; \text{andThen} \; (\text{String} \Rightarrow \text{AST}) = \text{identity}[\text{AST}]\]

• using this idea we get more than 3100 random tests which makes refactoring a lot easier

• powered by Simple Build Tool that I love!
jribble challenges

• language must encode a lot of information that normally compiler derives from the source code so it’s verbose

• grammar is complicated enough make parsing challenging

• using highly-functional approach makes it hard to reuse existing Java tools for profiling; we want to measure how long expressions are being evaluated and not how long method call takes
gwtc side

• in parallel to JDT we use jribble library to parse jribble files

• translate jribble AST nodes to GWT AST nodes

• fairly isolated with only a few, small extension points to existing gwtc
gwtc challenges

• programming in Java again, huh? ;-) 

• translation between ASTs is straightforward process but tiresome due to large number of cases 

• GWT’s mutable data structures makes it too easy to produce half-baked AST node that will confuse compiler and produce wrong output
Demo (advanced hello world)
Is there anything more?

• getting anything more advanced proved to be hard because there is no gradual progress

• anything more sophisticated involves Scala collections library that probably exploits every Scala language feature
Scala, Jribble and GWT

let’s recall our diagram

- scala standard library compiles and parses around 96% of files (~4400 jribble files)
- the remaining 4% percent has to be done carefully because it involves Unit, Nothing types and pattern matching logic
Showcase demo

- sizeable GWT app presenting many GWT features
- a lot of room for Scala to shine
- translation has been started and looks promising
- translated code heavily uses Scala collections so doesn’t work yet
Showcase in Scala

Avoid repetition by using higher-order functions and lambdas

Java

// Lists
TreeItem catLists = MainMenu.addItem(
    constants.categoryLists());
setupMainMenuOption(catLists,
    new CwListBox(constants),
    images.catLists());
setupMainMenuOption(catLists,
    new CwSuggestBox(constants),
    images.catLists());
setupMainMenuOption(catLists,
    new CwTree(constants),
    images.catLists());
setupMainMenuOption(catLists,
    new CwMenuBar(constants),
    images.catLists());
setupMainMenuOption(catLists,
    new CwStackPanel(constants),
    images.catLists());

Scala

// Lists
val catLists =
    MainMenu.addItem(constants.categoryLists)
    List(new CwListBox(constants),
        new CwSuggestBox(constants),
        new CwTree(constants),
        new CwMenuBar(constants),
        new CwStackPanel(constants)) foreach {
        setupMainMenuOption(catLists, _,
            images.catLists)
    }
Showcase in Scala

Java

```java
// Add a handler to handle drop box events
dropBox.addChangeHandler(new ChangeHandler() {
    public void onChange(ChangeEvent event) {
        showCategory(multiBox, dropBox.getSelectedIndex());
        multiBox.ensureDebugId("cwListBox-multiBox");
    }
};
```

Scala

```scala
// Add a handler to handle drop box events
dropBox.addChangeHandler { event: ChangeEvent =>
    showCategory(multiBox, dropBox.getSelectedIndex())
    multiBox.ensureDebugId("cwListBox-multiBox")
}
```

Get rid of anonymous classes by using lambdas and implicit conversions.
### Showcase in Scala

<table>
<thead>
<tr>
<th>Java</th>
<th>Scala</th>
</tr>
</thead>
<tbody>
<tr>
<td>`final HTML contactLink = new HTML(</td>
<td>`val contactLink: HTML = &lt;a href=&quot;javascript:undefined;&quot;&gt;</td>
</tr>
<tr>
<td>&quot;&lt;a href=&quot;javascript:undefined;&quot;&gt;&quot;</td>
<td>{contactName}&quot;&lt;/a&gt;</td>
</tr>
<tr>
<td>+ contactName + &quot;&lt;/a&gt;&quot;);</td>
<td>contactsPanel.add(contactLink);</td>
</tr>
<tr>
<td>contactsPanel.add(contactLink);</td>
<td></td>
</tr>
</tbody>
</table>

Instead of concatenating Strings use Scala’s XML literals that can be freely mixed with a code.

There can be more:

- Scala collections for data processing
- Wrappers around GWT APIs that make application code more idiomatic
The future

• Dev mode support has not been explored yet
• Getting showcase running
• Profiling and optimizing Jribble parser
• Merging extensions to gwtc and scalac upstream(?)
• More GWT samples being translated to Scala
Acknowledgements

• I’d like to thank many people that helped me with this project:
  • Rob Heittman
  • Aaron Novstrup
  • Lex Spoon

• I’d like to thank Google for opportunity to work on this project and for releasing so many projects on open source licenses.
Pointers

• Homepage:  http://scalagwt.gogoego.com/

• Groups:  http://groups.google.com/group/scalagwt

• Code review:  http://review.source.gogoego.com/

Questions?