

BTECH450

Flex/Silverlight Project

End of Year Semester

Academic Mentor: Xin Feng Ye

Industry Company: futuretech

Jason Wei-Lun Hsia



OVERVIEW

- ▶ Project goals
- ▶ RIA
 - ▶ Portability
 - ▶ Performance
 - ▶ Benchmarks
 - ▶ Multi-threading
 - ▶ Security
- ▶ Auto Code Generator
- ▶ Achievement
- ▶ Future work

PROJECT GOALS

- ▶ Explore and Research on the two RIA languages – Silverlight and Flex
- ▶ Define a set of guidelines and policies to enable the company to create the building blocks of a Silverlight version equivalent of the company's Flex based applications

WORK CARRIED OUT

- ▶ Usability
 - ▶ Portability
 - ▶ Performance
 - ▶ Security
 - ▶ Auto code generator
- 

PORTABILITY

▶ Silverlight

- ▶ Fully supported in Windows, Mac OS
- ▶ Available on Linux & Unix-based OS through 'Moonlight'
 - ▶ SUSE Linux Enterprise Desktop 11, openSUSE 11.x, Ubuntu 9.10, and Fedora 12
 - ▶ 'tracking' on the release of official Silverlight
- ▶ Supported on mobile devices ??? e.g. Windows Phone 7
 - ▶ Toolkits & Developer Tools available
 - ▶ But latest news says no

PORTABILITY (CONT'D)

▶ Flex

▶ Supported in almost all OS

- ▶ Windows, Mac OS 9/X, Linux, Solaris, HP-UX, Pocket PC/Windows CE, OS/2, QNX, Symbian, Palm OS, BeOS, and IRIX

▶ Adobe Flash Lite (light/mobile version of Flash) available for mobile devices

- ▶ Not on Apple devices (iPad, iPhone)

▶ Major mobile platforms (Apple & Windows Phone 7) are not supporting Silverlight and Flash

- ▶ Stability

- ▶ Security

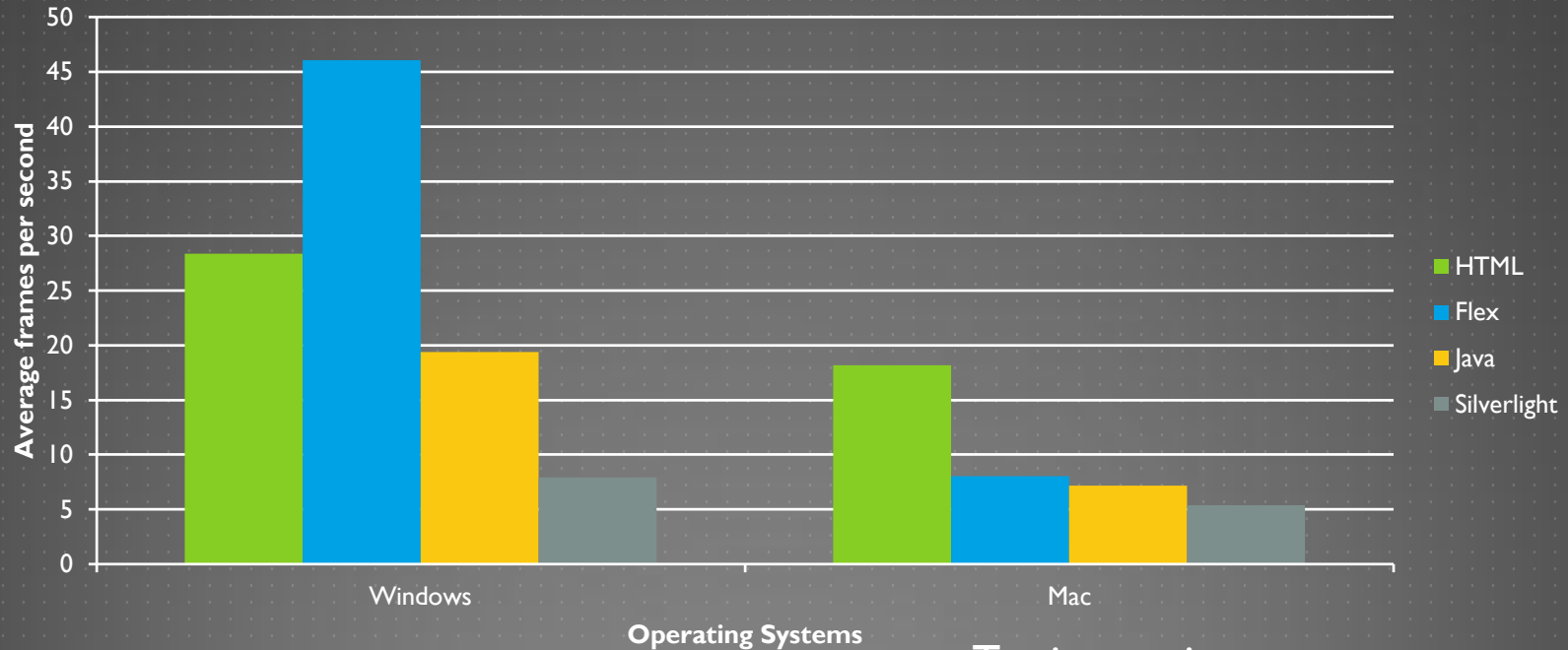
- ▶ Battery Life

PERFORMANCE

- ▶ Performance of RIA running on different environments
- ▶ Researched on RIA Benchmarking
 - ▶ GUIMark
 - ▶ BubbleMark
- ▶ Look into the graphics & multi-threading computation performance

GUIMARK RESULTS (2008)

Animation Benchmark of RIAs



Testing environment:

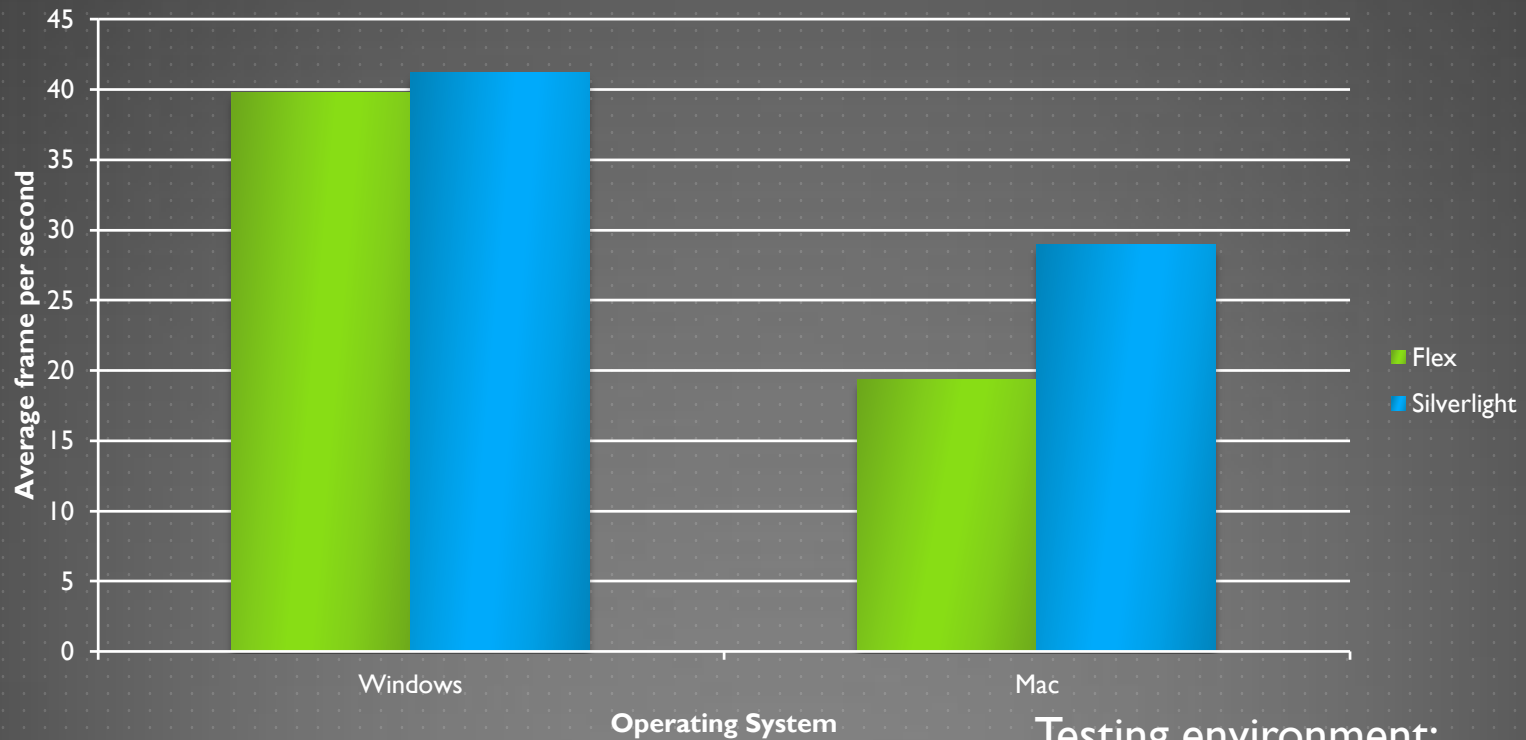
OS: Windows XP, Mac OS X 10.5

CPU: Intel Core 2 Duo 2.33 GHz

RIA: Flex 3, Silverlight 2

GUIMARK RESULTS (2010)

Animation Benchmarking for RIA



Testing environment:

OS: Windows 7, Mac OS X 10.6

CPU: Intel Core 2 Duo 2.4 GHz

RIA: Flex 4, Silverlight 4

LIMITATIONS & INTERESTING POINTS

- ▶ Only performance based on graphics
- ▶ Just a rendering tests
- ▶ Limited by Monitor's re-fresh rate
- ▶ Result varies largely between OS and Browsers

MULTI-THREADING

- ▶ Silverlight's attempt to enhance performance
 - ▶ Modern multi-core CPUs
- ▶ Flex lacks REAL multithreading

MULTI-THREADING

- ▶ Silverlight Multi-Thread Example
 - ▶ Based on BackgroundWorker
 - ▶ Separate from Silverlight application UI
 - ▶ Threads are created Explicitly
 - ▶ Identical to WPF program codes
- ▶ Flex
 - ▶ Single thread, no native support for multi-thread
 - ▶ Can be done through pseudo-thread

TESTING PROGRAM

- ▶ Based on nested loops Calculations
- ▶ Silverlight result collected from different number of threads
- ▶ Flex result only on single thread
- ▶ Testing environment
 - ▶ Quad-core CPU @ 2.33GHz
 - ▶ 4GB RAM
 - ▶ Windows 7 OS

BTECH450 Flex single Thread Test

Flex Single Thread Computation

Number of N: 5000000
Result: 348513
Time taken: 10.804 secs.

BTECH450 Silverlight Multi-Thread Test

Silverlight Multi-Thread Computation

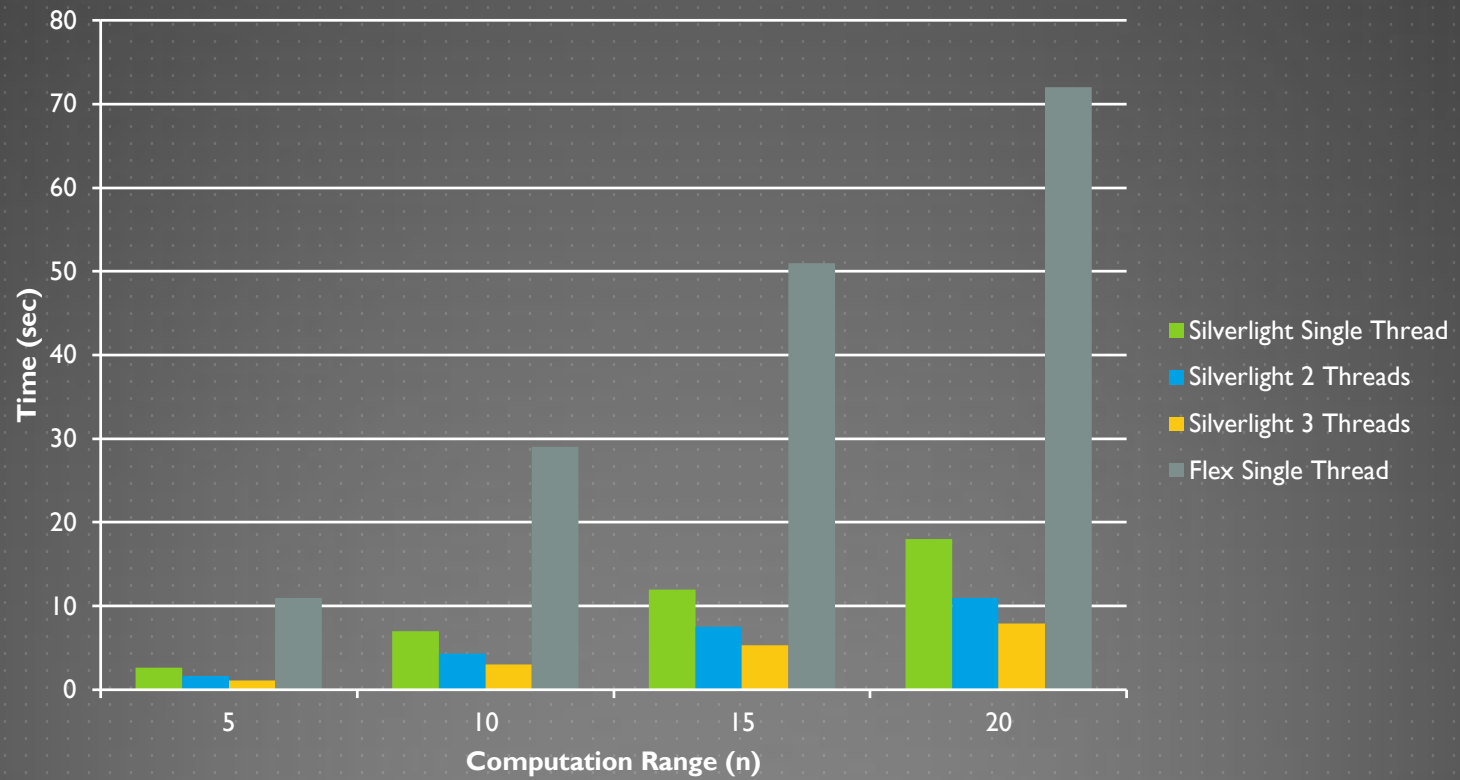
Single Thread

2 Threads

3 threads

Number of N: 5000000
Result: 0
Time taken: 1.1370651 secs.

Multithreading Computation



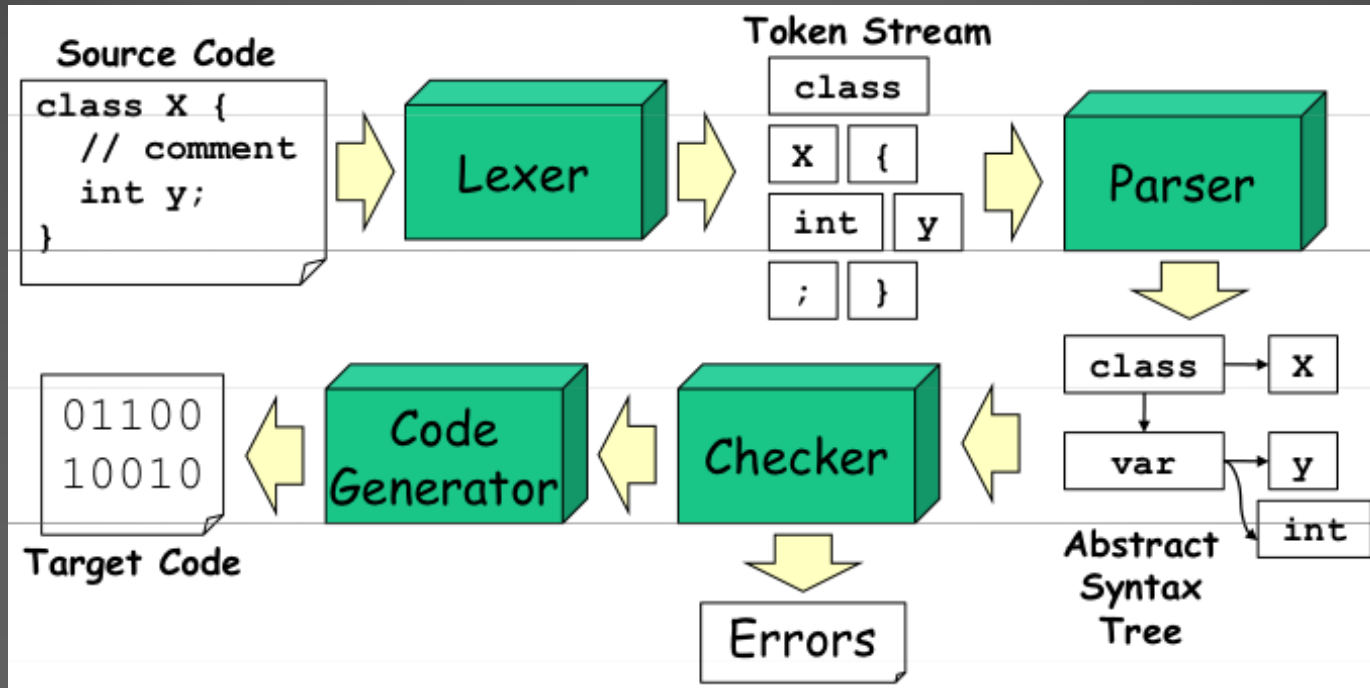
SECURITY

- ▶ Authentication & Authorisation provided by the RIA framework
- ▶ Silverlight
 - ▶ Same methods as ASP.NET
 - ▶ Authentication methods
 - ▶ Windows
 - ▶ Forms
 - ▶ Passport
- ▶ Flex
 - ▶ Authentication methods
 - ▶ Basic
 - ▶ Custom (form)
 - ▶ Additional Authentication provided by 3rd Party framework, e.g. Spring
 - ▶ Can be used with ASP.NET authentication features

AUTO CODE GENERATOR

- ▶ Normal string parsing can't achieve the goal
- ▶ Need a compiler tool
- ▶ Choice of
 - ▶ Coco/R
 - ▶ ANLTR

COMPILER TOOL



ANTLR

- ▶ Compiler Tool
- ▶ Written in Java
- ▶ Eclipse IDE environment
- ▶ Supported almost on every platform
- ▶ Generates Parser and Lexer based on an user-defined grammar

Java application

Grammar files

Generated
Parser and
Lexer

The screenshot displays an IDE interface with a project structure on the left and a code editor on the right. The project structure is organized into three main sections, each highlighted with a red rounded rectangle:

- src**: Contains a sub-package `a.b.c` with four Java files: `BTECH.java`, `Test1.java`, `Test2.java`, and `Test3.java`.
- Grammar files**: A collection of grammar files including `BTECH.g`, `BTECH.tokens [BTECH.g]`, `Sample.g`, `Sample.tokens [Sample.g]`, `Sample2.g`, `Sample2.tokens [Sample2.g]`, `Sample3.g`, `Sample3.tokens [Sample3.g]`, `Sample4.g`, and `Sample4.tokens [Sample4.g]`.
- Generated Parser and Lexer**: An `antlr-generated` sub-package containing generated Java files for each grammar: `BTECHLexer.java [BTECH.g]`, `BTECHParser.java [BTECH.g]`, `Sample2Lexer.java [Sample2.g]`, `Sample2Parser.java [Sample2.g]`, `Sample3Lexer.java [Sample3.g]`, `Sample3Parser.java [Sample3.g]`, `Sample4Lexer.java [Sample4.g]`, `Sample4Parser.java [Sample4.g]`, `SampleLexer.java [Sample.g]`, and `SampleParser.java [Sample.g]`.

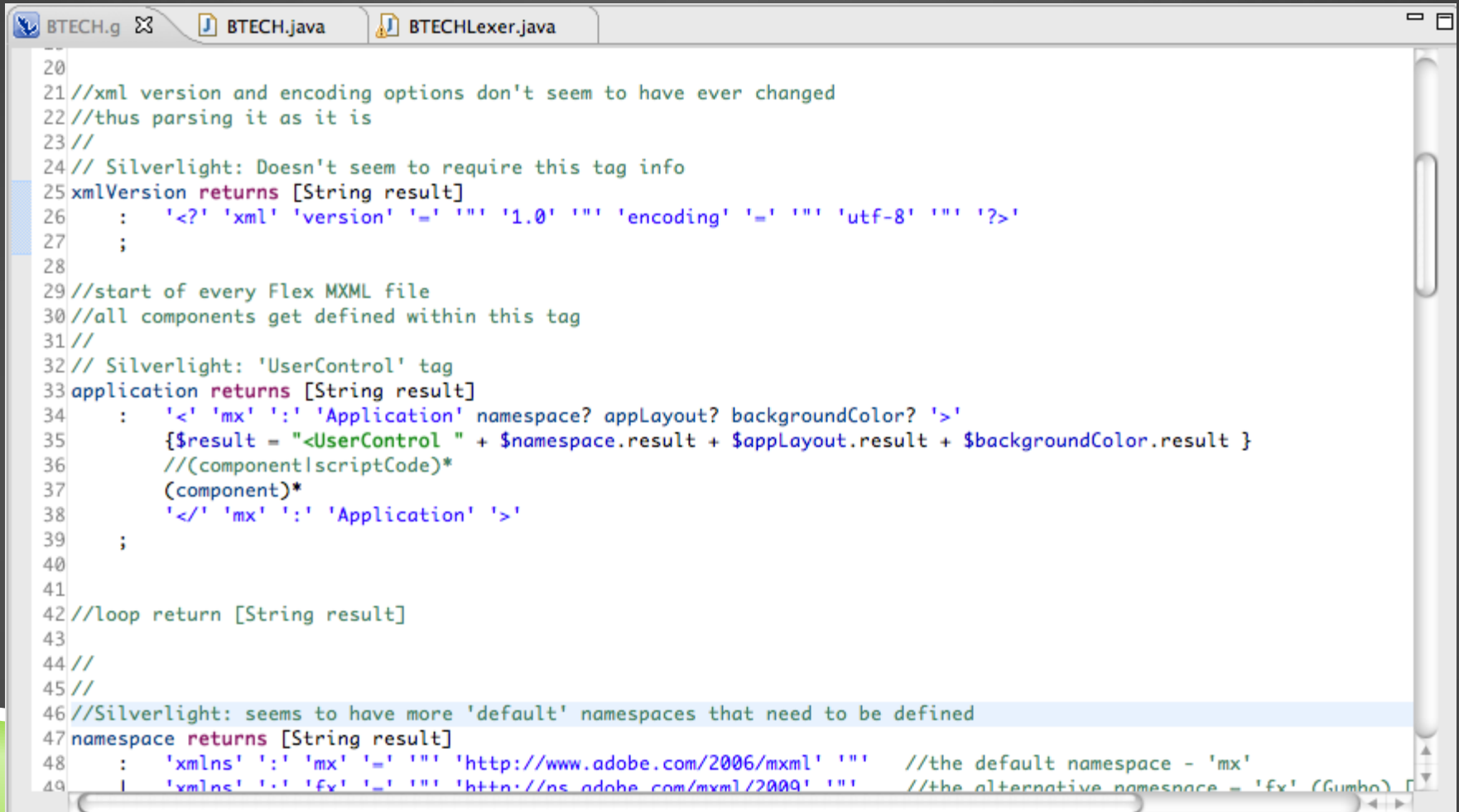
The code editor on the right shows a snippet of XML-like code with comments and tags, including:

```
24 // Silverlight: Doesn't s
25 xmlVersion returns [Strin
26 : '<?' 'xml' 'versi
27 ;
28
29 //start of every Flex MXM
30 //all components get defi
31 //
32 // Silverlight: 'UserCont
33 application returns [Stri
34 : '<' 'mx' ':' 'App
35 {$result = "<User
36 //(component|scri
37 (component)*
38 '</' 'mx' ':' 'Ap
39 ;
40
41
42 //loop return [String res
43
44 //
45 //
46 //Silverlight: seems to h
47 namespace returns [String
48 : 'xmlns' ':' 'mx'
49 | 'xmlns' ':' 'fx'
50 {
51 //xmlns="http://schem
```

At the bottom of the IDE, there are tabs for `Grammar`, `Interpreter`, and `Rail`. Below these is a `Problems` tab and a `Javadoc` tab. The `ANTLR Console` shows the following output:

```
ANTLR Parser Generator 3.2 Sep
Using project classpath: Yes.
Grammar: /Users/hsiajason/Docu
```

GRAMMAR



```
BTECH.g BTECH.java BTECHLexer.java
20
21 //xml version and encoding options don't seem to have ever changed
22 //thus parsing it as it is
23 //
24 // Silverlight: Doesn't seem to require this tag info
25 xmlVersion returns [String result]
26 :   '<?' 'xml' 'version' '=' ' "' '1.0' ' "' 'encoding' '=' ' "' 'utf-8' ' "' '?'>'
27 ;
28
29 //start of every Flex MXML file
30 //all components get defined within this tag
31 //
32 // Silverlight: 'UserControl' tag
33 application returns [String result]
34 :   '<' 'mx' ':' 'Application' namespace? appLayout? backgroundColor? '>'
35     {$result = "<UserControl " + $namespace.result + $appLayout.result + $backgroundColor.result }
36     //(component|scriptCode)*
37     (component)*
38     '</' 'mx' ':' 'Application' '>'
39 ;
40
41
42 //loop return [String result]
43
44 //
45 //
46 //Silverlight: seems to have more 'default' namespaces that need to be defined
47 namespace returns [String result]
48 :   'xmlns' ':' 'mx' '=' ' "' 'http://www.adobe.com/2006/mxml' ' "' //the default namespace - 'mx'
49 |   'xmlns' ':' 'fx' '=' ' "' 'http://ns.adobe.com/mxml/2009' ' "' //the alternative namespace - 'fx' (Gumbo)
```

GRAPHICAL REPRESENTATION (I)

Flex Source Code

```
for (i = 0 ; i < 10 ; i++) {  
total += i;  
}
```

Tokens

for

(

i

=

0

;

i

.....

}

GRAPHICAL REPRESENTATION (2)



Recognised by the parser this is a “for” loop rule

```
forLoop returns [String result]
:'for' '(' startCondition=assignmentStatement
endCondition=comparisonStatement
increment=variableIncrement ')'
'{'

{$result+= "for" + "(" + $startCondition.result +
$endCondition.result + $increment.result + ")" + "{";}
(' anyStatement=statement {$result+= $anyStatement.result;})*
'}'
{$result+= "}";}
;
```

Grammar Rule



GRAPHICAL REPRESENTATION (3)



Equivalent Silverlight Code

```
for (i = 0 ; i < 10 ; i++) {  
    total += i;  
}
```

ACHIEVEMENTS

- ▶ Knowledge in RIA & the frameworks (Silverlight & Flex)
- ▶ Guidelines & comparisons between the basic controls and code structure in Silverlight and Flex
- ▶ Better understanding in different aspects of RIA
 - ▶ Portability
 - ▶ Performance
 - ▶ Usability
 - ▶ Security
- ▶ Auto code generator development

FUTURE WORKS

- ▶ More complete work on auto code generator
 - ▶ Such as error detection on the semantic meaning of code
- ▶ Research on other aspects of RIA technologies & HTML5

THANK YOU