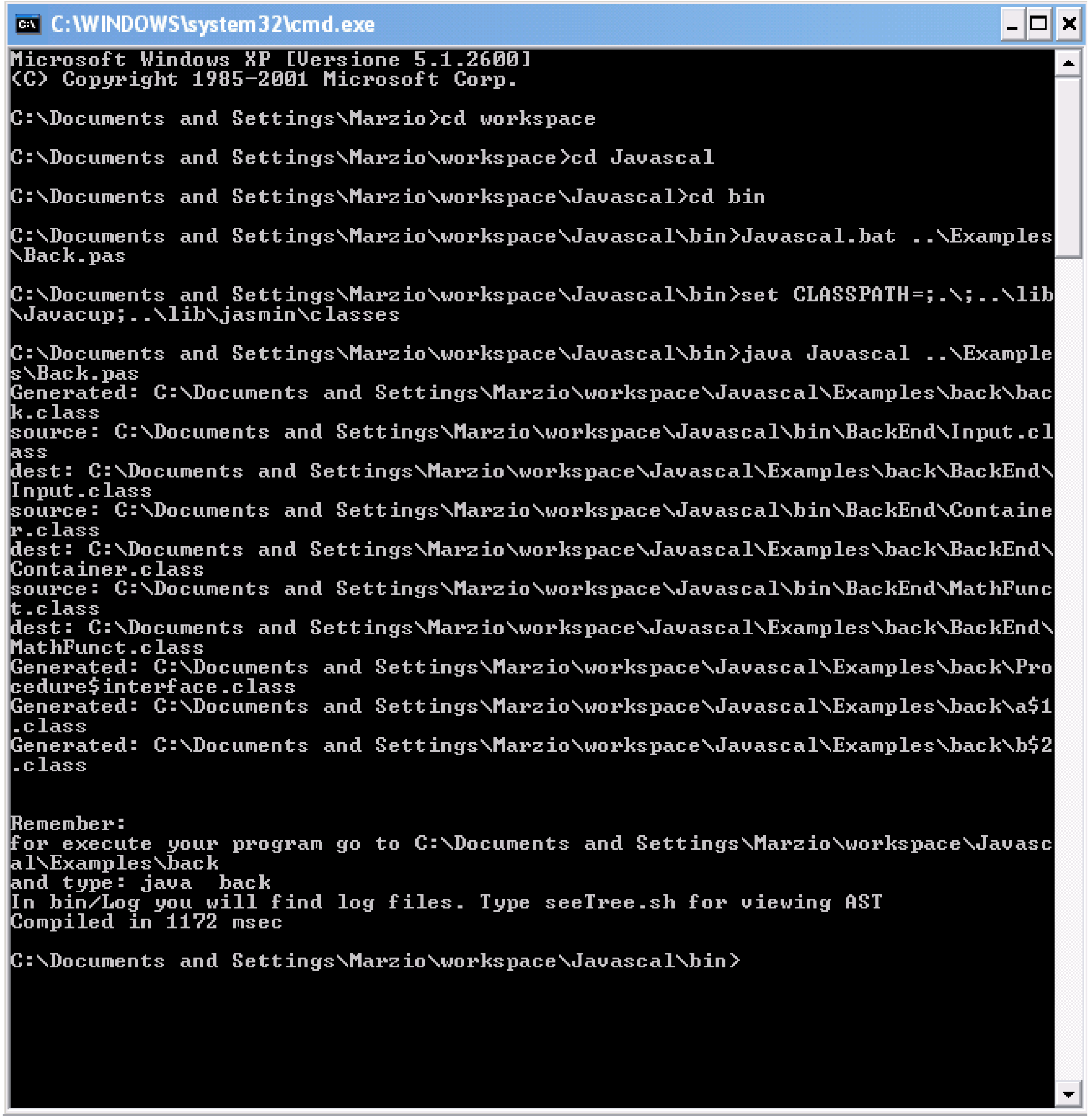
Running Javascal

Before running Javascal ensure to have compiled Javascal:

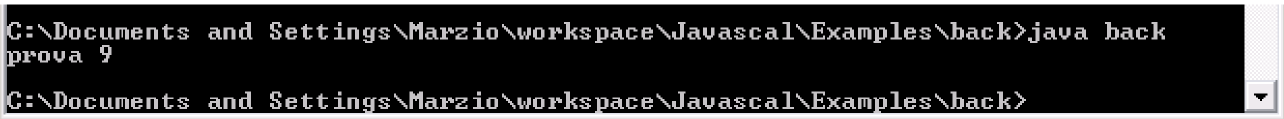
- following the Getting Started With Javascal – Developers Guide or
- downloding the compiled version from javascal sourceforge download page.

1 – Open Command Prompt and go into **Javascal\bin** folder
2 – Compile a sample pascal application typing: **Javascal.bat ..\Examples\Back.pas**



```
Microsoft Windows XP [Versione 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Marzio>cd workspace

C:\Documents and Settings\Marzio\workspace>cd Javascal

C:\Documents and Settings\Marzio\workspace\Javascal>cd bin

C:\Documents and Settings\Marzio\workspace\Javascal\bin>Javascal.bat ..\Examples
\Back.pas

C:\Documents and Settings\Marzio\workspace\Javascal\bin>set CLASSPATH=;.\;..\lib
\Javacup;..\lib\jasmin\classes

C:\Documents and Settings\Marzio\workspace\Javascal\bin>java Javascal ..\Example
s\Back.pas
Generated: C:\Documents and Settings\Marzio\workspace\Javascal\Examples\back\bac
k.class
source: C:\Documents and Settings\Marzio\workspace\Javascal\bin\BackEnd\Input.cl
ass
dest: C:\Documents and Settings\Marzio\workspace\Javascal\Examples\back\BackEnd\
Input.class
source: C:\Documents and Settings\Marzio\workspace\Javascal\bin\BackEnd\Containe
r.class
dest: C:\Documents and Settings\Marzio\workspace\Javascal\Examples\back\BackEnd\
Container.class
source: C:\Documents and Settings\Marzio\workspace\Javascal\bin\BackEnd\MathFunc
t.class
dest: C:\Documents and Settings\Marzio\workspace\Javascal\Examples\back\BackEnd\
MathFunct.class
Generated: C:\Documents and Settings\Marzio\workspace\Javascal\Examples\back\Pro
cedure$interface.class
Generated: C:\Documents and Settings\Marzio\workspace\Javascal\Examples\back\a$1
.class
Generated: C:\Documents and Settings\Marzio\workspace\Javascal\Examples\back\b$2
.class


Remember:
for execute your program go to C:\Documents and Settings\Marzio\workspace\Javasc
al\Examples\back
and type: java  back
In bin/Log you will find log files. Type seeTree.sh for viewing AST
Compiled in 1172 msec

C:\Documents and Settings\Marzio\workspace\Javascal\bin>
```

3 – Now you can run the compiled ByteCode:



```
C:\Documents and Settings\Marzio\workspace\Javascal\Examples\back>java back
prova 9

C:\Documents and Settings\Marzio\workspace\Javascal\Examples\back>
```

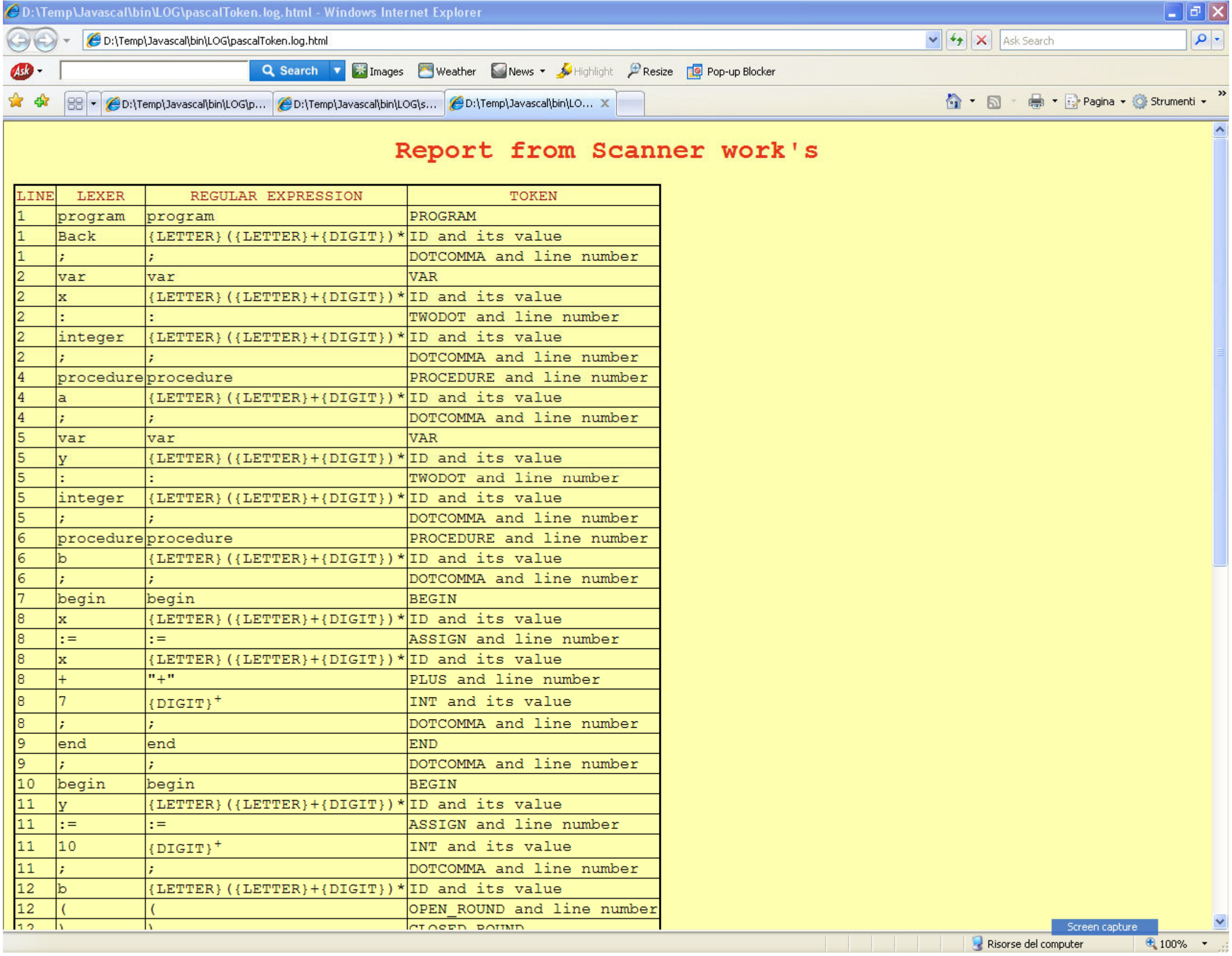
See LOG produced by Javascal

Every time you compile a Pascal source file Javascal creates a lot of useful information. In particular you can see:

- The **syntax tokens** recognized by the scanner
- The **semantic phrases** recognized by the parser
- The **symbol table** produced during the parser
- The **abstract syntax tree** produced after running assembler
- The **assembly** files produces by the assembler.

Syntax tokens

You can see the syntax tokens opening the file **bin\LOG\pascalToken.log.html**

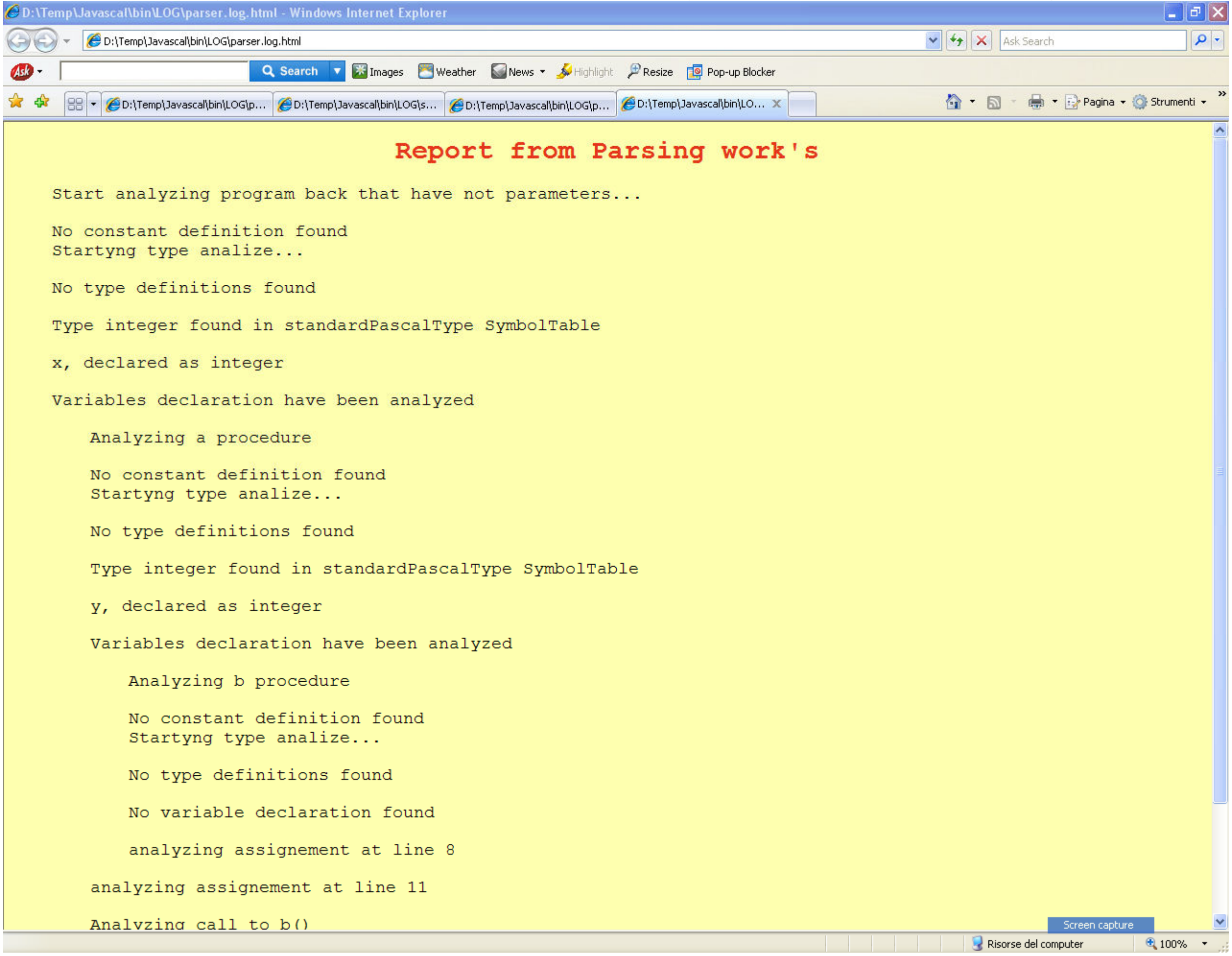

D:\Temp\Javascal\bin\LOG\pascalToken.log.html - Windows Internet Explorer
D:\Temp\Javascal\bin\LOG\pascalToken.log.html
Ask Search
Search
Images
Weather
News
Highlight
Resize
Pop-up Blocker
D:\Temp\Javascal\bin\LOG\p...
D:\Temp\Javascal\bin\LOG\s...
D:\Temp\Javascal\bin\LO...
Pagina
Strumenti
Screen capture
Risorse del computer
100%

Report from Scanner work's

LINE	LEXER	REGULAR EXPRESSION	TOKEN
1	program	program	PROGRAM
1	Back	{LETTER}({LETTER}+{DIGIT})*	ID and its value
1	;	;	DOTCOMMA and line number
2	var	var	VAR
2	x	{LETTER}({LETTER}+{DIGIT})*	ID and its value
2	:	:	TWODOT and line number
2	integer	{LETTER}({LETTER}+{DIGIT})*	ID and its value
2	;	;	DOTCOMMA and line number
4	procedure	procedure	PROCEDURE and line number
4	a	{LETTER}({LETTER}+{DIGIT})*	ID and its value
4	;	;	DOTCOMMA and line number
5	var	var	VAR
5	y	{LETTER}({LETTER}+{DIGIT})*	ID and its value
5	:	:	TWODOT and line number
5	integer	{LETTER}({LETTER}+{DIGIT})*	ID and its value
5	;	;	DOTCOMMA and line number
6	procedure	procedure	PROCEDURE and line number
6	b	{LETTER}({LETTER}+{DIGIT})*	ID and its value
6	;	;	DOTCOMMA and line number
7	begin	begin	BEGIN
8	x	{LETTER}({LETTER}+{DIGIT})*	ID and its value
8	:=	:=	ASSIGN and line number
8	x	{LETTER}({LETTER}+{DIGIT})*	ID and its value
8	+	"+"	PLUS and line number
8	7	{DIGIT}$^+$	INT and its value
8	;	;	DOTCOMMA and line number
9	end	end	END
9	;	;	DOTCOMMA and line number
10	begin	begin	BEGIN
11	y	{LETTER}({LETTER}+{DIGIT})*	ID and its value
11	:=	:=	ASSIGN and line number
11	10	{DIGIT}$^+$	INT and its value
11	;	;	DOTCOMMA and line number
12	b	{LETTER}({LETTER}+{DIGIT})*	ID and its value
12	(	(	OPEN_ROUND and line number
13	)	)	CLOSED_ROUND

Semantic phrases

You can see the syntax recognized by the parser opening the file **bin\LOG\parser.log.html**

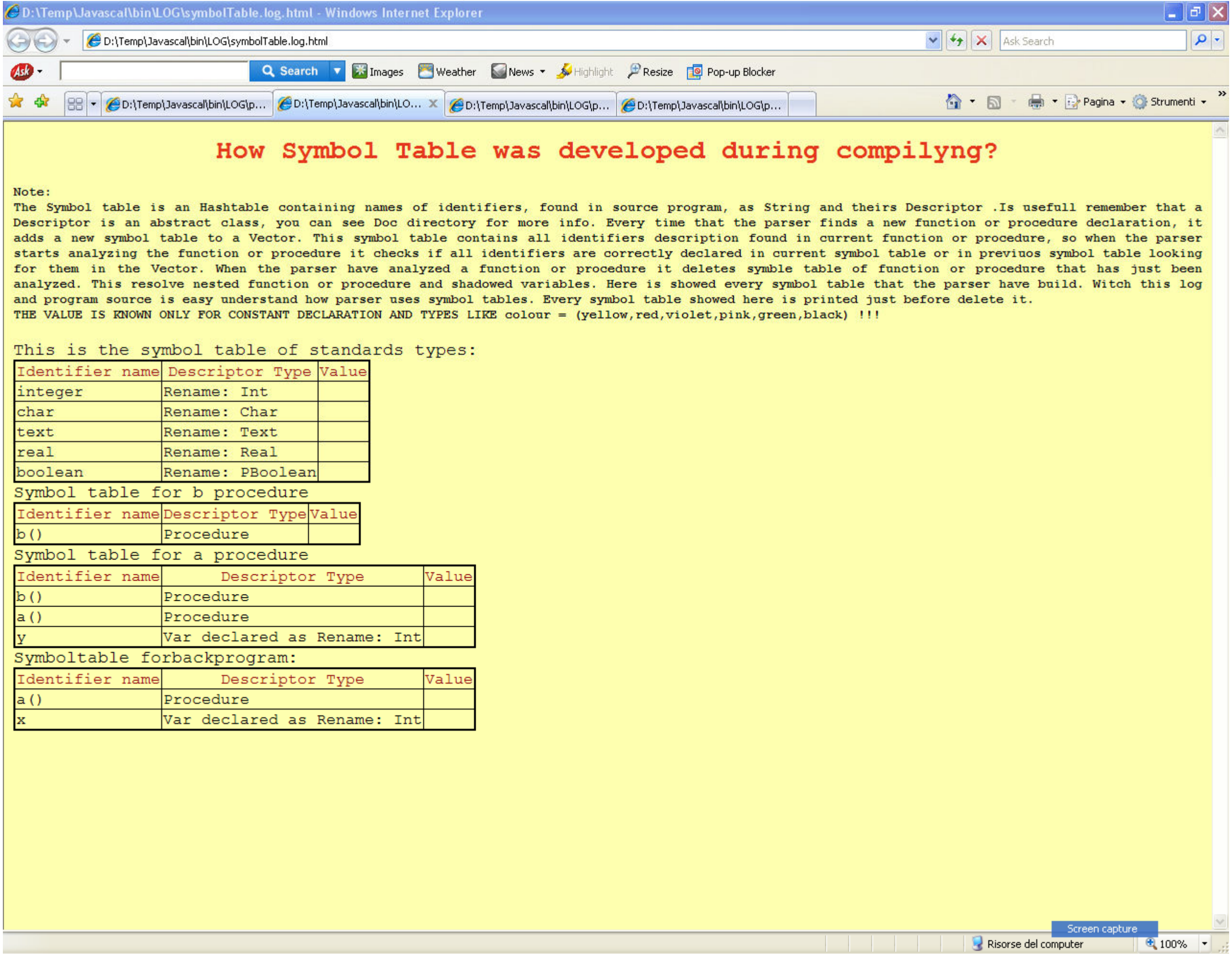

D:\Temp\Javascal\bin\LOG\parser.log.html - Windows Internet Explorer
D:\Temp\Javascal\bin\LOG\parser.log.html
Ask Search
Search
Images
Weather
News
Highlight
Resize
Pop-up Blocker
D:\Temp\Javascal\bin\LOG\p...
D:\Temp\Javascal\bin\LOG\s...
D:\Temp\Javascal\bin\LOG\p...
D:\Temp\Javascal\bin\LO...
Pagina
Strumenti
Screen capture
Risorse del computer
100%

Report from Parsing work's

```
Start analyzing program back that have not parameters...

No constant definition found
Startyng type analize...

No type definitions found

Type integer found in standardPascalType SymbolTable

x, declared as integer

Variables declaration have been analyzed

    Analyzing a procedure

    No constant definition found
    Startyng type analize...

    No type definitions found

    Type integer found in standardPascalType SymbolTable

    y, declared as integer

    Variables declaration have been analyzed

        Analyzing b procedure

        No constant definition found
        Startyng type analize...

        No type definitions found

        No variable declaration found

        analyzing assignement at line 8

    analyzing assignement at line 11

    Analyzing call to b()
```

Symbol table

You can see the Symbol table opening the file **bin\LOG\ symbolTable.log.html**


D:\Temp\Javascal\bin\LOG\symbolTable.log.html - Windows Internet Explorer
D:\Temp\Javascal\bin\LOG\symbolTable.log.html
Ask Search
Search
Images
Weather
News
Highlight
Resize
Pop-up Blocker
D:\Temp\Javascal\bin\LOG\p...
D:\Temp\Javascal\bin\LO...
D:\Temp\Javascal\bin\LOG\p...
D:\Temp\Javascal\bin\LOG\p...
Pagina
Strumenti
Screen capture
Risorse del computer
100%

How Symbol Table was developed during compilyng?

Note:
The Symbol table is an Hashtable containing names of identifiers, found in source program, as String and theirs Descriptor .Is usefull remember that a Descriptor is an abstract class, you can see Doc directory for more info. Every time that the parser finds a new function or procedure declaration, it adds a new symbol table to a Vector. This symbol table contains all identifiers description found in current function or procedure, so when the parser starts analyzing the function or procedure it checks if all identifiers are correctly declared in current symbol table or in previuos symbol table looking for them in the Vector. When the parser have analyzed a function or procedure it deletes symble table of function or procedure that has just been analyzed. This resolve nested function or procedure and shadowed variables. Here is showed every symbol table that the parser have build. Witch this log and program source is easy understand how parser uses symbol tables. Every symbol table showed here is printed just before delete it.
THE VALUE IS KNOWN ONLY FOR CONSTANT DECLARATION AND TYPES LIKE colour = (yellow,red,violet,pink,green,black) !!!

This is the symbol table of standards types:

Identifier name	Descriptor Type	Value
integer	Rename: Int	
char	Rename: Char	
text	Rename: Text	
real	Rename: Real	
boolean	Rename: PBoolean	

Symbol table for b procedure

Identifier name	Descriptor Type	Value
b()	Procedure	

Symbol table for a procedure

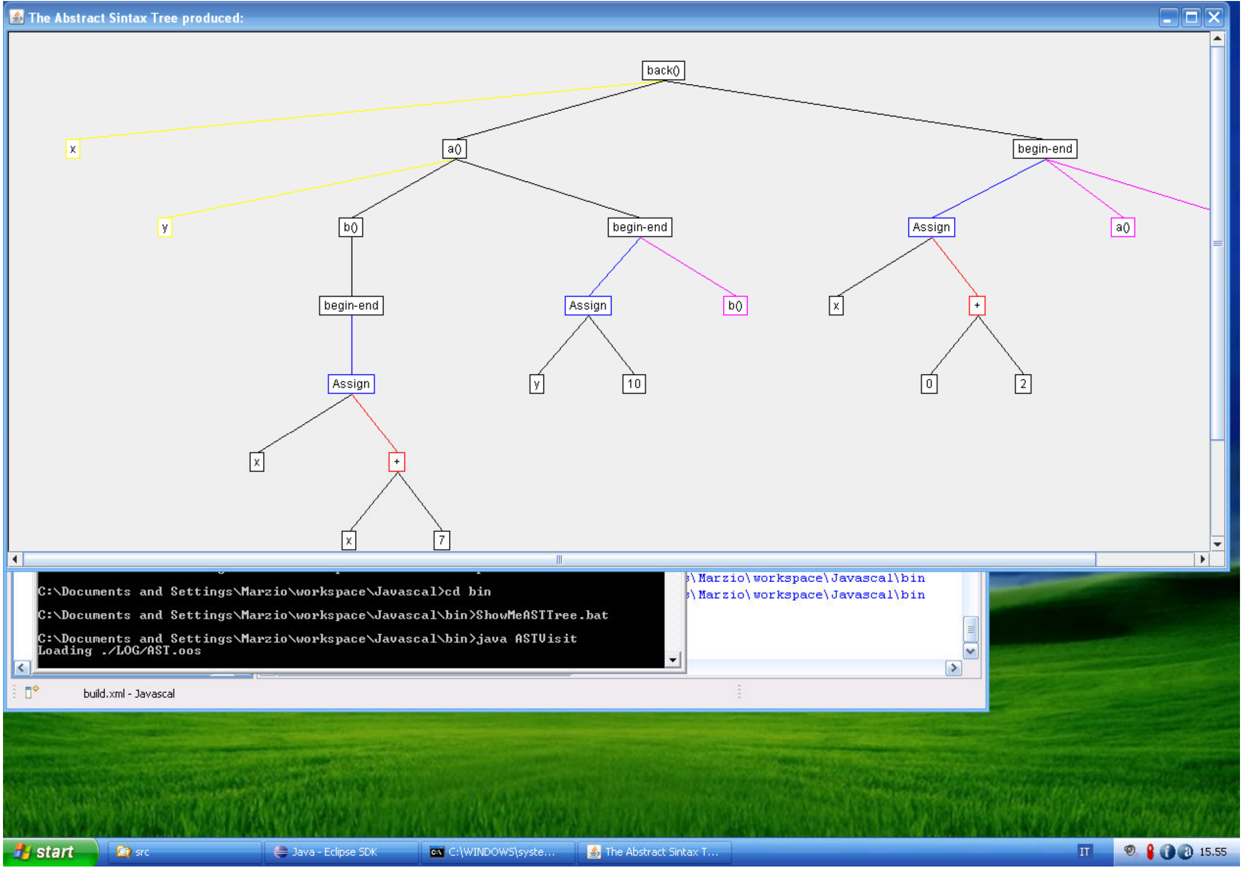
Identifier name	Descriptor Type	Value
b()	Procedure	
a()	Procedure	
y	Var declared as Rename: Int	

Symboltable forbackprogram:

Identifier name	Descriptor Type	Value
a()	Procedure	
x	Var declared as Rename: Int	

Abstract syntax tree

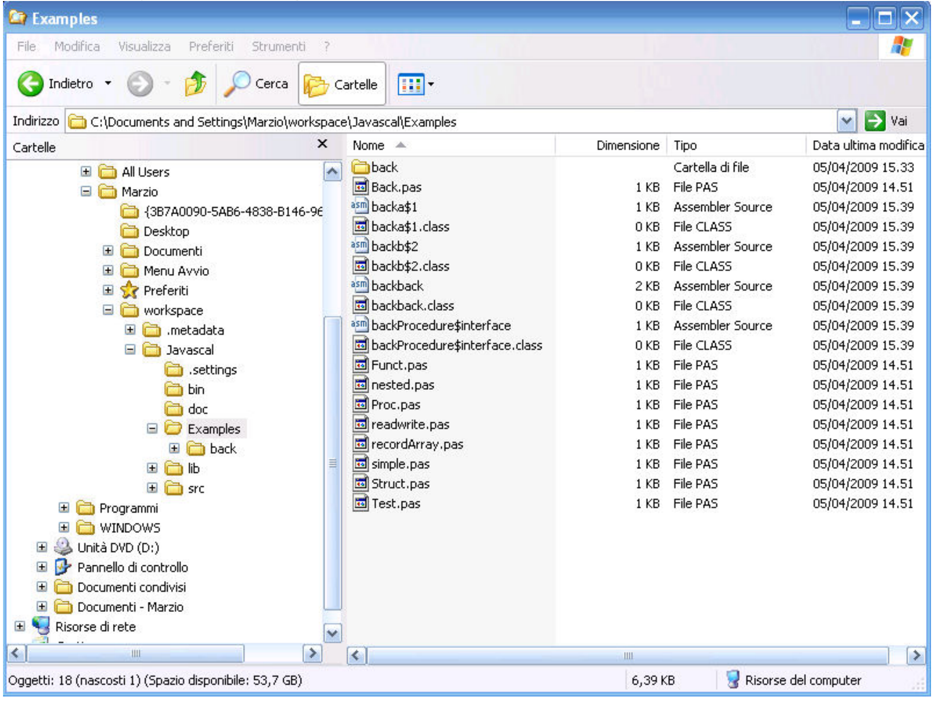
You can see the AST running the **ShowMeASTTree.bat** located into **Javascal\bin** folder



```
C:\Documents and Settings\Marzio\workspace\Javascal>cd bin

C:\Documents and Settings\Marzio\workspace\Javascal\bin>ShowMeASTTree.bat

C:\Documents and Settings\Marzio\workspace\Javascal\bin>java ASTVisit
Loading ./LOG/AST.oos
```

Assembly

You can see the assembly files taking a look into Examples folder (.asm files)



Nome	Dimensione	Tipo	Data ultima modifica
back		Cartella di file	05/04/2009 15.33
Back.pas	1 KB	File PAS	05/04/2009 14.51
backa$1	1 KB	Assembler Source	05/04/2009 15.39
backa$1.class	0 KB	File CLASS	05/04/2009 15.39
backb$2	1 KB	Assembler Source	05/04/2009 15.39
backb$2.class	0 KB	File CLASS	05/04/2009 15.39
backback	2 KB	Assembler Source	05/04/2009 15.39
backback.class	0 KB	File CLASS	05/04/2009 15.39
backProcedure$interface	1 KB	Assembler Source	05/04/2009 15.39
backProcedure$interface.class	0 KB	File CLASS	05/04/2009 15.39
Funct.pas	1 KB	File PAS	05/04/2009 14.51
nested.pas	1 KB	File PAS	05/04/2009 14.51
Proc.pas	1 KB	File PAS	05/04/2009 14.51
readwrite.pas	1 KB	File PAS	05/04/2009 14.51
recordArray.pas	1 KB	File PAS	05/04/2009 14.51
simple.pas	1 KB	File PAS	05/04/2009 14.51
Struct.pas	1 KB	File PAS	05/04/2009 14.51
Test.pas	1 KB	File PAS	05/04/2009 14.51

Oggetti: 18 (nascosti 1) (Spazio disponibile: 53,7 GB) 6,39 KB Risorse del computer