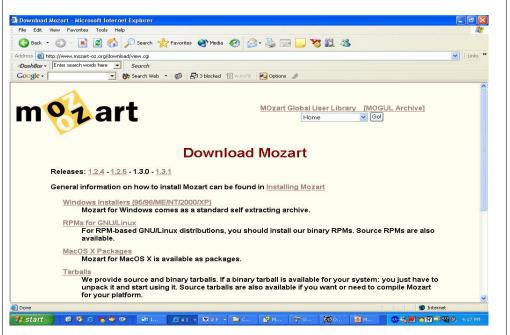
Programming Language Concepts, CS2104 export PATH Lab/Assignment 0 (Lab Session, 3-6pm, 24th Aug 2007) (b) just type the following command in the command line: Deadline for Lab0 : 5pm 28Aug 2007 Tue (submit via IVLE) export PATH=\$PATH:/home/course/cs2104/mozart/bin CS2104 is a 4 credit points module (written final exam 50%, 2 midterm exams 25%, 3. To run Mozart from sunfire, you may need X-Window to be installed on your lab/tutorial assignments 25%). The module homepage is Windows machine. To install X-Window, please use the guide from the following web http://www.comp.nus.edu.sg/~cs2104 and IVLE. Teaching means lectures and combined page: https://www.comp.nus.edu.sg/cf/x/index.html tutorials/lab sessions (labs). Lectures are based of the book: ■ Peter Van Roy, Seif Haridi: Concepts, Model, and Techniques of Computer **Running Mozart on sunfire** Programming, The MIT Press, 2004 1. start x-win32 on your windows machine. The main purposes of tutorials are: for self-assessment, revise material from lectures, 2. connect to sunfire using a ssh client answer questions, allow deep understanding, prepare labs assignments. Tutorials 3. type "oz&" in the command line. comprise simple assignments, and are good exercises for the exam. You may discuss tutorials/chapters on the IVLE discussion groups. There will be five lab assignments Kev Bindings (please submit in time). C-. C-1 Feed current line C-. C-r Feed selected region **Overview** C-. C-b Feed whole buffer ■ This lab/assignment should be done individually. C-. C-p Feed current paragraph At the end of lab/assignment, you should have Mozart running on your computer. C-. c Toggle display of *Oz Compiler* buffer Try the examples that have been introduced in the first lecture together with some C-. e Toggle display of *Oz Emulator* buffer similar functions. positions the transcript to make the first error message C-x ' (i.e. Control-x backquote) ■ Use the time available to ask questions! visible and moves the point, in the source buffer, to Ask your friends. where the bug is likely to be located. You can also ask on IVLE's discussion group of Chapter 1. C-. n Create a new buffer using the Oz major mode. Note that this buffer has no associated file name, so Useful Software quitting Emacs will kill it without warning. http://www.mozart-oz.org/ M-n □ programming language: Oz M-p Switch to the previous resp. next buffer in the buffer □ system: Mozart (1.3.0, released on April 15, 2004) list that runs in an Oz mode. If no such buffer exists. □ interactive system an error is signalled. Requires Emacs on your computer (http://www.gnu.org/software/emacs/) Available from module webpage For more details about Mozart commands, you should consult Programming ■ First tutorial will help with installation Environment and Tools manual. For more details about emacs commands, you should consult the Emacs on-line tutorial available from the Help menu in the Emacs menu bar Mozart Installation (Windows/Unix) or an online tutorial from http://www.lib.uchicago.edu/keith/tcl-course/emacstutorial.html. **Details for the Windows Installation** Install Emacs and Mozart on your PC (very easily). **Details for the Unix Installation** 1. Get an account on sunfire. **2.** Add to your PATH the following new path: /home/course/cs2104/mozart/bin. You can do this in two ways either (a) or (b): (a) modify your ".profile" or ".bashrc" file such that the file will contain the following two commands: PATH=\$PATH:/home/course/cs2104/mozart/bin



Emacs Installation

http://www.gnu.org/software/emacs/windows/

6.2.7 Emacs

You need an installed Emacs. You can get it at http://www.gnu.org/software/emacs/windows/. Unpack it somewhere, for example to /cygdrive/c/program Files/ and execute the addpm. exe binary in the bin subdirectory. We will refer to the directory where Emacs is installed as *emacs*.

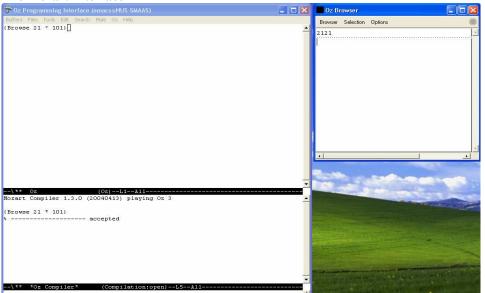
The Mozart System

- Interactive interface (the declare statement)
 - Allows introducing program fragments incrementally and execute them
 - □ Has a tool (Browser), which allows looking into the store using the procedure Browse
- {Browse 21 * 101} -> by selecting "Oz" panel, "Feed Line" or alternatively "C-. C-l", this will display in the Browser window the number 2121

Running our first Oz program

Kuilling Our	mst Oz program					
Microsoft PowerPoi	1 New Office Document		1	DJ Java Decompiler 3.4		RealPlayer
: Elle Edit View	Open Office Document		-	InstallShield for Microsoft Visual C++ 6		🍣 Yahoo! Mail
	💕 Windows Catalog		12	Microsoft Visio		
	🌯 Windows Update		1	Creative	E	
6	💷 WinZip			Ulead Photo Express 4.0 My Custom Edition	F	
	Set Program Access and Defaults		w	WordWeb		
	Mahoo! Messenger		1	Cumulus	F	
7 Martin Ma Martin Martin Mart	Accessories	•	1	dBpowerAMP Music Converter	F	
	Microsoft Office Tools		1	Lexmark	ы	
	MUSNET-IV		1	MATLAB 6.5	ы	
	CuickTime		1	Nikon View 4	F	
Andrei Stefan	m Real		1	Ordix Mpack Pro	F	
Andrei Steran	m Startup	<u> </u>		PrintMe Internet Printing	E	
Internet	m WinZip		111	Residential Listings 2003 - 2004	F	
	Internet Explorer			Trend Micro OfficeScan Client	F.	
E-mail	3 Outlook Express		1	X-Win32 5.4	•	
Outlook Express	Remote Assistance				ы	
Notepad	Windows Media Player		1	Adobe Reader 6.0		
	3 Windows Messenger			XviD	•	
F-Secure SSH Client	F-Secure SSH Client		m	AC3Filter	×.	
	GHC GHC		1 4	BSPlayer		
Acrobat Reader 5.0	6 Ghostgum			DashBar	۲	
	m Ghostscript			Date Manager	۲	
DVI Viewer	m Hugs98		(TT	DIVX		
GSview 4.3	🛅 Java Web Start		1	GAIN Publishing	F	
	Creator LE		m	GNU Emacs	F.	
Microsoft PowerPoint	mathematica 4.1		1	Microsoft Office	F	
	Microsoft Visual C++ 6.0				×	Documentation
All Programs ▶	miktex				•	🤹 Oz Programming Interface
	i Objective Caml		-	TortoiseCVS	•	An integrated development environment for
	C WIDRAR	•		Yahoo! Messenger	۲	
🛃 start 🔰 🚳 🕯	m XWin32 ver 5.3	•	A	Acrobat Reader 5.0		
🐉 start 🔰 🙆 🍯			100	Adobe Photoshop Album 2.0 Starter Edition		

The Mozart Interface



Concept of (Single-Assignment) Variable Identifier

declare X = 21

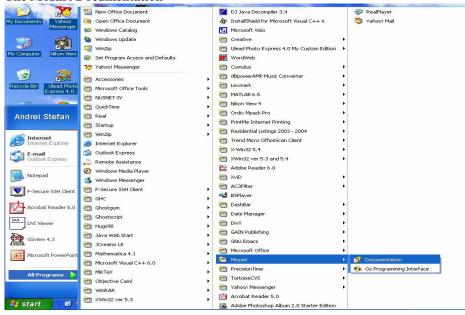
```
X = 22
% raise an error
X = 21
% do nothing
declare
X = 22
% from now on, X will be bound to 22
```

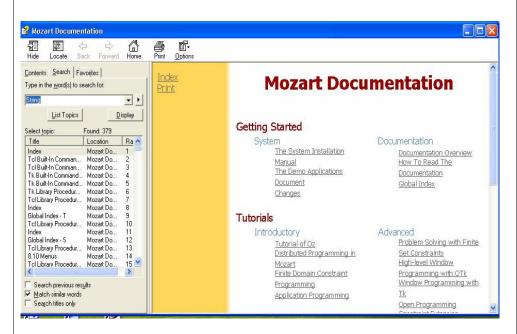
Concept of Oz Variable Type

A variable type is known only after the variable is bound Examples: $1. \times < 1$ $\times < 1.0$

```
2.declare X Y
X = "Oz Language"
Y = 'Oz Language'
if X == Y
then {Browse yes}
else {Browse no}
end
```

The Mozart Documentation





Concept of Oz Variable Type

declare X Y Z
X = "Oz Language"
Y = 'Oz Language'
{String.toAtom X Z}
if Z == Y then {Browse yes}
else {Browse no}
end

StringToAtom

{String.toAtom +5 ?A}

converts a string s to an atom A. s must not contain NUL characters. This is the inverse of Atom.tostring (which see).

Try these Functions

declare
fun {Minus X}
 ~X
end
{Browse {Minus 15}}
declare
fun {Max X Y}
 if X>Y then X else Y end
end
declare

```
X = {Max 22 18}
Y = {Max X 43}
{Browse Y}
```

Exercise 1 (Absolute Value) Write a function Abs that computes the absolute value of a number. This should work for both integers and real numbers.

Try Recursive Function

```
Recursive function definition
fun {Fact N}
    if N == 0 then 1
    else N * {Fact N-1}
    end
end
{Browse {Fact 5}}
```

Try some calls:

- {Fact 5}
- {Fact 100}
- {Fact 10000} Use the Oz Panel to get an idea how much memory is needed.

Oz Panel

	🖉 Oz Panel				
fun (Fact N)	Panel Options				
<pre>if N == 0 then 1 else N * {Fact N - 1} end end (Browse (Fact 10000))</pre>	Threads Memory Problem Solving)			
(1000)	Runtime				
	Run: 0.73 s 🔳	1.0			
	Garbage Collection: 0.44 s 🔳	0.8			
	Copy: 0.00 s	0.4			
		0.2			
	Propagation: 0.00 s 🔳				
** fact.oz (Oz)L12Top					
fun {Fact N}	Threads				
if N == 0 then 1 else N * {Fact N - 1} end	Created: 476	6			
end	Runnable: 1	4			
{Browse {Fact 10000}}	rumasie.	2			
Declared variables:		1			
Fact: procedure/2		0			
\$ accepted					
\localSwitches		Contract Contraction Contract on the Contract of Contract			
\switch +threadedqueries -verbose -expression -r	unwithdebugger	A ALL AND AND AND A A			
{Panel.open}					
\$ accepted		and a subscription of the second s			

Try Fibonacci Example

The execution time of a program as a function of input size, up to a constant factor, is called the program's **time complexity**.

```
declare
fun {Fibo N}
case N of
1 then 1
[] 2 then 1
```

```
[] M then {Fibo (M-1)} + {Fibo (M-2)}
end
end
{Browse {Fibo 100}}
```

The time complexity of {Fibo N} is proportional to 2^{N} .

Try Efficient Fibonacci Example

```
declare
fun {FiboTwo N A1 A2}
    case N of
        1 then A1
    [] 2 then A2
    [] M then {FiboTwo (M-1) A2 (A1+A2)}
    end
end
{Browse {FiboTwo 100 1 1}}
```

The time complexity of $\{Fibo N\}$ is proportional to N.

Exercise 2 (Power) Compute n^m where n is an integer and m is a natural number. **Hint**: Use the following inductive definition of nm:

```
n<sup>0</sup> = 1
n<sup>m</sup> = n * n<sup>m-1</sup>
Write a function Pow as follows:
declare
fun {Pow N M}
if ... then
...
else
...
end
end
```

Exercise 3 (Maximum Recursively) Compute the maximum of two natural numbers, knowing that the only allowed test with a conditional is the test whether a number is zero (that is, if N==0 then ... else ... end).

Hint: Facts about the maximum $(n \ge 0 \text{ and } m \ge 0)$:

- $\blacksquare max(n, m)=m, if n=0.$
- $\blacksquare \max(n, m) = n, \text{ if } m = 0.$
- $\blacksquare \max(n, m) = 1 + \max(n-1, m-1), \text{ otherwise.}$