McGill University
School of Computer Science
COMP 520

Wig Compiler Report

Report No. yyyy-xx

First Member
Second Member
Third Member

November 21, 2002

Contents

List of Figures

List of Tables

1 Introduction

1.1 Clarifications

Have you discovered any unclear points in the WIG language definition? How have you chosen to resolve these?

1.2 Restrictions

Have you deliberately made any restrictions in your version of WIG? What was your motivation? What are the implications?

1.3 Extensions

Have you made any extensions to your version of WIG? What was your motivation? What are the implications?

1.4 Implementation Status

What is the status of your WIG implementation? Have all your proposed features been implemented? Have they been tested? Do they work?

2 Parsing and Abstract Syntax Trees

2.1 The Grammar

Give the full grammar for your version of the WIG language by listing your `bison` input with all actions removed.

2.2 Using the `flex` or `SableCC` Tool

Discuss any interesting points in your `flex` implementation of the scanner. What are your token kinds? Did you use start conditions? How and why?

2.3 Using the `bison` or `SableCC` Tool

Discuss any interesting points in your `bison` implementation of the parser. How did you make `bison` accept your grammar?

2.4 Abstract Syntax Trees

Present the structure of your abstract syntax trees. If you used `SableCC`, discuss how you modified the tree.

2.5 Desugaring

Do you resolve any syntactic sugar during parsing? How and why?

2.6 Weeding

Do you weed unwanted parse trees? How and why?

2.7 Testing

How have you tested this phase? Does it work?

3 Symbol Tables

3.1 Scope Rules

Describe the scope rules of your language.

3.2 Symbol Data

Describe the contents of symbol table entries.

3.3 Algorithm

Describe your algorithm for building symbol tables and checking scope rules.

3.4 Testing

How have you tested this phase? Does it work?

4 Type Checking

4.1 Types

Describe the types supported by your language.

4.2 Type Rules

Describe the type rules of your language.

4.3 Algorithm

Describe your algorithm for checking type rules.

4.4 Testing

How have you tested this phase? Does it work?

5 Resource Computation

5.1 Resources

Describe the resources that you compute.

5.2 Algorithm

Describe your algorithm for computing resources.

5.3 Testing

How have you tested this phase? Does it work?

6 Code Generation

6.1 Strategy

Describe the overall strategy for generating code for a service.

6.2 Code Templates

Describe the code templates for your language constructs.

6.3 Algorithm

Describe your algorithm for generating code.

6.4 Runtime System

Describe the runtime system that is used by the services you generate.

6.5 Sample Code

Show the complete code generated for the service `tiny.wig`.

6.6 Testing

How have you tested this phase? Does it work?

7 Availability and Group Dynamics

7.1 Manual

Describe how to compile and install services.

7.2 Demo Site

Give the URL for a web site that contains demos of services that you have generated.

7.3 Division of Group Duties

Explain who worked on what parts of the compiler and how you divided the work.