

Index

(Index is still under construction)

Symbols

`%{,%}` 62

A

Accessor functions 42

 C++ 48

Adding member functions 44

 naming convention 46

`%addmethods` directive 44, 54, 230

`%alpha` directive 78

`%apply` directive 94

Arrays 31, 39

 and pointers 35

 Tcl 233, 257

 Typemaps 105

 use with structures. 43

ASCII 74

ASCII Documentation module 86

ASCII strings 32

Automatic documentation generation 18

Automatic documentation generation (See
 Documentation system)

B

`bool` datatype 32

C

C

 Ambiguity of data types 16

 Built in datatypes 31

 Constructors 43

 Destructors 43

 Exception handling 109–111

 Exception handling with `longjmp()` 110

 Integers 32

 Macros 30

 Parsing source files 14

 Preprocessor 30

`-c` option 29

C++ 48–55

 Accessor functions 48

 Adding default constructors and destructors
 44

 Adding new methods to classes 54

 Constants 50

 Constructors 49

 Destructors 49

 Documentation 89

 Dynamic loading 267

 Example 48

 Exception handling 111

 Inheritance 51

 Inheritance and multiple modules 263

 Interface building strategy 66

 Interface to classes 24

 Member data 49

 Member functions 49–??

 Multiple inheritance 51

 Operator overloading 31

 Parsing source files 14, 31, 55

 Partial class definitions 55

 Protection 50

 Read only members 50

 References 50

 Relation to SWIG pointer handling 48

 Renaming classes 54

 Renaming members 53

 Static member functions 49

 Supported features 48

 Templates 31

 Unsupported features 48

`-c++` option 29, 49

Call by reference

 overriding 40

Call by value 37

`char *` datatype 32

 Use within structures 43

`char` datatype

 Example 33

Character strings 32

Checking for NULL pointers 108

Classes

 and pointers 36

 Tcl 222

 with enum 50

Code insertion 61–63

- Code insertion block % {,% } 30, 62
- Code reuse 17
- Combination C/SWIG header files 14
- Comments 30
 - And documentation 15
 - Documentation system 78
 - Ignoring 79
- Compilation problems
 - Tcl 217
- Complex datatypes 36
- Conditional compilation 14, 59
 - Expression evaluation 61
- const
 - Creating constants 34
- Constants 24, 34
 - Expressions 34
 - Function pointers 41
 - Tcl 221
- Constraints 107
- Constructors 43, 49
 - for C programs 43
- Cross platform development 19

D

- D option 29, 60
- d option 29
- dascii option 29, 75
- Data types
 - Built-in 31
 - Floating point 32
 - Integer truncation 32
 - Integers 32
 - Strings 32
- Default arguments 40
 - and ANSI C 41
- Default constructors and destructors 44
- #define 30, 34
- Destructors 43, 49
 - for C programs 43
- Developer Studio 163
- Developer studio 118
- dhtml option 29, 75
- Directives 30
- %disabledoc directive 80
- dlatex option 29, 75
- DLL (Dynamic Link Library)

- Tcl 218
- dnone option 29, 75
- Documentation
 - Automatic generation 18
 - Comments 15
 - Example 15
- Documentation system 74–89
 - Adding text 79
 - Argument names 75
 - C++ 89
 - Comment handling 78
 - Default formatting 77
 - Documentation entry 74
 - Example 80–85
 - Format variables 77
 - Formats 75
 - Formatting 76
 - Functions 75
 - Ignoring comments 79
 - Information strings 79
 - Limitations 74
 - Sections 75
 - Sorting 78
 - Specifying formatting styles 76
 - Tab expansion 79
 - Titles 75
- double datatype 32
- Dynamic loading 27, 266
 - Irix 27
 - Irix 5.3 161
 - Linux 27
 - Solaris 27
 - Tcl 216

E

- #elif 30, 59
- #else 30, 59
- embed.i library file 161
- %enabledoc directive 80
- #endif 30, 59
- enum 34
 - In classes 50
- Event driven programming 17
- Event loop 17
- %except directive 109, 238
- Exception handling

- Tcl 238
- Exceptions 109–114
 - \$except variable 114
 - \$function variable 109
 - C programs 109–111
 - C++ 111
 - Debugging 114
 - Defining multiple handlers 112
 - Deleting 109
 - except typemap method 112
 - Perl 5 110
 - Python 109
 - Using longjmp() 110
- Expressions 34
- Extending 271–309
 - Accessor function generation 277
 - C++ compiler 272
 - C++ processing 304
 - Cleanup 295
 - Code generation functions 289
 - Command creation 295
 - Command line options 291
 - Compiling a new module 272
 - Constants 299
 - DataType class 279
 - Default typemaps 300
 - Documentation system 306
 - File management 288
 - Function parameters 282
 - Future directions 309
 - Global variables 298
 - Hash tables 284
 - Header files and support code 292
 - Language class 273–276
 - main() program 272
 - Module naming 294
 - Naming services 289
 - Output format 273
 - ParmList class 283
 - Parsing 292
 - Pointer type checker 293
 - Required files 272
 - Sample language module 290–299
 - String class 283
 - SWIG Library 303
 - SWIG Organization 271
 - Typemap API 286
 - Typemap lookup procedure 287
 - Wrapper functions 296
 - WrapperFunction class 285
- %extern directive 68
- F**
 - FILE * data type 36
 - float datatype 32
 - Floating point 32
 - Format variables 77
 - Fortran 40
 - Function pointers 31
 - Functions
 - call by value 37
 - Default arguments 40
 - Optional arguments 40
 - Pointers 41
 - renaming 39
 - return by value 38
- G**
 - gd library 178
 - Global variables
 - Tcl 220
 - globals option 165
 - guile option 29
- H**
 - help option 29
 - Helper functions 62, 98
 - htcl option 215
 - htk option 215
 - HTML 74
 - HTML Documentation module 86–88
- I**
 - I option 29, 69
 - #if 30, 59
 - #ifdef 30, 59
 - #ifndef 30, 59
 - Ignoring comments 79
 - %import directive 69
 - %include directive 68
 - Including files 69
 - Inheritance 51

- Example 51
- %init directive 63
- Initialization code 63
- %inline directive 63
- Inlined code 63
- Input format 29
- int datatype 32
- Integers
 - 32 bit 32
 - 64 bit 32
 - Truncation 32
- Interface building 63–67
 - ANSI C/C++ 64
 - C++ 66
 - Header files 64
 - Interface file 64
 - Interface from hell 66
 - main() program 64
 - Preparing a C program 64
 - Preprocessor directives 64
 - Problematic declarations 64
 - Reducing code size 66
 - Rules of thumb 66
- Interface file 13, 29, 64

L

- l option 29, 217
- LaTeX 74
- LaTeX Documentation module 88–89
- Linux
 - Dynamic loading 216
- Lists
 - and pointers 35
- %localstyle directive 77
- long datatype 32
- long double datatype 32

M

- Macintosh 19
- main() function
 - Interface building
 - main() 65
- make_default option 44
- make_default pragma 44
- MATLAB 138–143
- Member data 49

- Accessor functions 49
 - Read only 50
- Member functions 49–??
- Memory leaks
 - Return by value 38
- Modular programming 17
- %module directive 30, 73
- module directive 215
- module option 29
- Multiple inheritance 51
- Multiple modules 263

N

- %name directive 39, 53
- Namespace conflicts 40
 - namespace option 215, 218
- %native directive 207
- Nested structures 46
 - Splitting 46
- Netscape plugin 260–261
- NMAKE 219
 - Perl 5 119
 - Python 163
- no_default pragma 44
- noobject option 215, 226
- NULL pointer 35, 108
 - Tcl 221

O

- o option 29, 30
- objc option 29
- Object oriented programming
 - C Programs 42
- OpenGL 234–238
- Optional arguments 40
 - and ANSI C 41

P

- package option 120
- Parser
 - Built in datatypes 31
 - Constants 31
 - Functions 31
 - Limitations 16, 30
 - Variables 31
- Parsing

- C++ source 55
- Perl 5 115–159
 - Accessing array structure members 148
 - ActiveWare port 118
 - Array access 127
 - Blessed references 121
 - C++ classes 122
 - C++ compilation 117
 - Checking for NULL pointers 125
 - Classes 120
 - Compiling example 13
 - Compiling with NMAKE 119
 - Constants 121
 - Converting char ** to a Perl array. 146
 - Default type conversions 151
 - Directed graph example 123–137
 - Dynamic modules 116
 - Exception handling 143
 - Exceptions 110
 - Floating point conversion functions 150
 - Function wrappers 120
 - Global variables 121
 - Header files 115
 - Inheritance 157
 - Inheritance example 51
 - Integer conversion functions 150
 - Interface file 124
 - Iterators 158
 - MATLAB example 138–143
 - Modules 120
 - Nested objects 156
 - newSViv() function 150
 - NULL pointer 122
 - Object ownership 155
 - package option 120
 - Packages 120
 - perlmain.i library file 117
 - Pointer handling functions 152
 - Pointers 121
 - Rapid prototyping 130
 - Reference functions 151
 - References vs. pointers 122
 - Return values 152
 - Returning function arguments 147
 - Shadow classes 26, 47, 131, 153–159
 - Shadow functions 157
 - Static linking 116
 - String conversion functions 151
 - Structures 122, 123
 - sv_setiv() function 150
 - sv_setnv() function 150
 - SvIOK() function 150
 - SvIV() function 150
 - SvNV() function 150
 - Typemap example 108
 - Typemaps 144–152
 - Windows 95/NT 118
 - Wrapping C libraries 138
- perl4 option 29
- perl5 option 29, 115
- perlmain.i library file 117
- plugin option 215
- Pointers 34–37
 - and arrays 35
 - and lists 35
 - C++ References 50
 - classes 36
 - NULL 35
 - Run time type checking 35
 - Scripting language representation 35
 - String representation 35
 - structures 36
 - Tcl 221
 - To built-in types 34
 - to Functions 41
 - unions 36
- Pointers to functions 31
- Predefined symbols 61
- prefix option 215, 218
- Preprocessor 30
 - Macros 30
- Python
 - Accessing array structure members 198
 - Accessing arrays 173
 - Adding member functions 180
 - Arrays 174, 176
 - Built-in exceptions 192
 - C++ classes 167
 - C++ modules 162
 - Callback functions 203–206
 - Compilation problems 162
 - Compiling example 14

- Constants 166
- Constructors 209
- Converting char ** 194
- Destructors 209
- Dynamic modules 161
- embed.i library file 161
- Exception handling 190
- Exceptions 109
- File conversion functions 200
- File pointers 196
- Floating point conversion functions 199
- Functions 164
- gd library 178
- Global variables 165
- globals option 165
- GRAD Extension 169
- Header files 160
- Heat equation example 170–178
- Imaging class 181–184
- Implementing native methods 206
- Importing modules 162
- Inheritance 212
- Integer conversion functions 199
- List conversion functions 199
- MESS extension 169
- Module names 208
- Module naming 161
- Modules 164
- Nested objects 210
- NULL pointer 166
- Object ownership 209
- Passing small arrays 197
- Performance concerns 213
- Pointers 166, 202
- Preparing C libraries 178
- Rebuilding the Python interpreter 161
- Returning function arguments 196
- Shadow classes 26, 45, 54, 168–170, 207–214
- Shadow functions 209
- shadow option 169
- Standard typemaps 200
- Static linking 161, 188
- String conversion functions 199
- Structures 166
- this pointer 208

- Tuple conversion functions 200
- Typemap example 108
- Typemaps 193–??
- Using multiple modules 188
- Variable linking 165
- Windows 95/NT 162
- python option 29, 160

R

- %raw directive 78
- Read only variables 39
- %readonly directive 39, 50
- %readwrite directive 39, 50
- %rename directive 40
- Renaming 39
 - C++ members 53
 - Class names 54
- Return by value 38
- Returning arguments
 - Tcl 244
- Runtime library 265

S

- Scripting Language
 - Code Management 22
- Scripting Languages 21
 - C/C++ programming. 21
 - Constants 24
 - Parsing 22
 - Performance 22
 - Shadow classes 24
 - Two language model 21
 - Variable linking 24
 - Wrapper functions 23
- %section directive 76
- Shadow classes
 - Defined 24
 - Perl 5 26, 47, 131
 - Python 26, 45, 54
 - Tcl 26, 259
- Shared libraries 27, 266
- short datatype 32
- signed char datatype 32
- signed datatype 32
- Static initialization of modules 73
- Static linking 26

- Tcl 216
- Static member functions 49
- static option 116
- strict option 35
- Strings 32
 - Embedded NULL bytes 32
- Structures 42–47
 - Accessor functions 42
 - adding member functions 44
 - and pointers 36
 - Array members 43
 - char * members 43
 - Constructors 43
 - Defining 42, 47
 - Destructors 43
 - Nested 46
 - Splitting 46
 - Tcl 221
- %style directive 77
- %subsection directive 76
- %subsubsection directive 76
- SWIG
 - Bug reporting 9
 - Cross platform development 19
 - Directives 30
 - FTP site 7
 - Macintosh 19
 - Mailing list 7
 - New features 9
 - Organization 12
 - Parsing limitations 16
 - Web page 7
 - Windows 95/NT 19
- swig command 13, 29
- SWIG Library 68–73
 - Creating library files 71
 - Defined 69
 - Overriding built-in files 71
 - Searching order 69
- SWIG/C Header file 59
- SWIG_GetPointerObj function 251
- SWIG_GetPtr function 251
- SWIG_LIB Environment variable 69
- swig_lib file 69
- SWIG_MakePtr function
 - SWIG_SetPointerObj function 251

Symbols

- Defining 60
- Predefined 61

T

Tcl

- Arrays 233, 257
- Binary trees 226–231
- BLT 255
- C++ classes 222
- cget method for objects 224
- char ** to list conversion 241
- Combining extensions 255
- Compilation problems 217
- Compiling example 13
- Configuration management 251–257
- configure method for objects 225
- Constant handling with typemaps 243
- Constants 221
- Conversion to structures to lists 245
- Creating objects 223
- Data structures and Tk 231
- Deleting objects 224
- Directed graph example 231–233
- Dynamic loading 216, 256
- Example 31
- Exception handling 238
- Expect 254
- expect.i library 253
- Global variables 220
- Headers and libraries 215
- ish.i library 253
- iTcl namespaces 218
- itclsh.i library 253
- itkwish.i library 253
- iwish.i library 253
- Loading of dynamic modules 216
- Member data 224
- Member functions 224
- Netscape plugin 260–261
- Object oriented interface 223–226
- Objects and pointers 225
- OpenGL example 234–238
- Package prefix 217
- Packages 256
- Performance and objects 226

- Pointers 221, 250
- Rebuilding tclsh 216
- Rebuilding wish 216, 217
- Returning arguments 244
- Sample wrapper function 23
- Shadow classes 26, 259
- Standard typemaps 249
- Static linking 216
- Structures 221
- SWIG_RcFileName symbol 253
- Tcl 8.0 262
- Tcl_AppendElement 247
- Tcl_AppendResult 246
- Tcl_AppInit 252
- Tcl_GetDouble 246
- Tcl_GetInt 246
- Tcl_Merge 247
- Tcl_PrintDouble 246
- Tcl_SetDoubleObj 247
- Tcl_SetResult 246
- Tcl_SplitList 247
- tclsh.i library 216, 252
- Tix 255
- Typemaps 239–251
- Variable Linking 24
- Windows 95/NT 119, 163, 218, 219
- wish.i library 216, 252
- Wrapper functions 220
- Tcl 8.0
 - Pointers 221
 - Tcl_DecrRefCount 249
 - Tcl_DuplicateObj 249
 - Tcl_GetDoubleFromObj 247
 - Tcl_GetIntFromObj 247
 - Tcl_GetStringFromObj 248
 - Tcl_IncrRefCount 249
 - Tcl_IsShared 249
 - Tcl_ListObjAppendElement 248
 - Tcl_ListObjAppendList 248
 - Tcl_ListObjGetElements 248
 - Tcl_ListObjIndex 248
 - Tcl_ListObjLength 248
 - Tcl_ListObjReplace 248
 - Tcl_NewDoubleObj 247
 - Tcl_NewIntObj 247
 - Tcl_NewListObj 248
 - Tcl_NewObj 249
 - Tcl_NewStringObj 247
 - Tcl_SetIntObj 247
 - Tcl_SetStringObj 247
 - Tcl_StringObjAppend 248
- tcl option 29, 215
- tcl8 option 29, 215
- tclsh.i library file 70
- %text directive 79
- %title directive 76
- Type casting 269
- Type checking 35, 264, 268
 - strictness setting 35
- Type checking performance 270
- typedef 31, 37, 41
 - and structures 42
 - Pointers to functions 41
- %typedef directive 41
- %typemap directive 99
- Typemaps 90–??
 - \$dim variable 106
 - \$mangle variable 105
 - \$source variable 105
 - \$target variable 105
 - \$type variable 105
 - \$value variable 105
 - ANY keyword 106
 - argout method 102
 - Arrays 101, 105
 - check method 102, 107
 - Common methods 102
 - const method 102
 - Copying 101
 - Creating 100
 - default method 103
 - Deleting 101
 - Example 99, 106
 - except method 112
 - freearg method 102
 - ignore method 103
 - in method 102
 - Matching rules 101
 - memberin method 102, 106
 - memberout method 102
 - Motivation 98
 - Multidimensional arrays 106

Named 100
 out method 102
 ret method 102
Scope 104
Special variables 105
SWIG Library 107
Tcl 239–251
 varin method 102
 varout method 102
typemaps
 arginit method 103

U

Unions 42–47
 Accessor functions 42
 and pointers 36
Unknown datatypes 36
unsigned char datatype 32
unsigned datatype 32
unsigned long datatype 32
unsigned short datatype 32

V

-v option 29
%val directive 40
Variable length arguments 31
Variable Linking
 Access through function calls. 24
 Defined 24
 Direct access 24
 Tcl 24
Variable linking
 to complex datatypes 38
Variables
 Read only 39
 renaming 39
-version option 29
Visual C++ 218
 Developer studio 218

W

Windows 95/NT 19
 Perl 5 118
 Python 162
 Tcl 218
%wrapper directive 63

Wrapper functions 22
 Tcl 220