

OBJECTIVE-C FOUNDATION CLASSES

REFERENCE CARD

Part 5: Compound classes

DArguments

Methods

- **init** Init an argument parser
- **deepen** Not implemented
- **free** Free the parser, incl. options
- **option** : (char *) longOpt : (char *) shortOpt
 | : (char *) descr : (id) target Add an option
- **options** : (DArgOption *) options : (int) nr
 | Add multiple options
- **parse** : (char *) name : (char *) usage : (char *) version
 | : (char *) tail : (char **) argv : (int) argc
 | Parse and process the arguments
- **printVersion** : (char *) version Print version info
- **printOptionHelp** : (char *) shortOpt : (int) longCol
 | : (char *) longOpt : (char *) descr Print option help
- **printHelp** : (char *) usage : (char *) tail ... Print all help

DCalendar

Constants

- DCL_SUNDAY Sunday
- DCL_MONDAY Monday
- DCL_TUESDAY Tuesday
- DCL_WEDNESDAY Wednesday
- DCL_THURSDAY Thursday
- DCL_FRIDAY Friday
- DCL_SATURDAY Saturday
- DCL_ALL_MONTHS All months in calendar

Methods

- + (int) **firstWeekDay** Return first day of week
- + **firstWeekDay** (int) day Set first day of week
- + (BOOL) **isLeapYear** : (int) year Test for leap year
- + (int) **leapYears** : (int) from : (int) to ... Count leap years
- + (int) **daysInMonth** : (int) year : (int) month
 | Return number of days in month
- + (int) **weekDay** : (int) year : (int) month : (int) day
 | Return day of week
- **init** Init empty calendar
- **init** : (int) year Init calendar with year
- **init** : (int) year : (int) month Init calendar with

- | year, month
- (int) **year** Return current year
- **year** : (int) year Set current year
- (int) **month** Return current month
- **month** : (int) month Set current month
- (DData *) **toData** Get calendar in new data string
- (DText *) **toText** Get calendar in new text string

DConfigReader

Methods

- **init** Init config reader
- **free** Free config reader
- (BOOL) **parse** : (id) source : (char *) name : (id) handler
 | Parse the source with name, results to handler

DConfigWriter

Methods

- **init** Init config writer
- **init** : (id) dest Init config writer with config dest.
- **free** Free the config writer
- (BOOL) **startConfig** : (id) dest Start writing config
- (BOOL) **startConfig** Start writing config
- (BOOL) **endConfig** Done writing config
- (BOOL) **section** : (char *) name Write section in config
- (BOOL) **option** : (char *) sect : (char *) opt
 | : (char *) val Write option in config
- (BOOL) **comment** : (char *) comment Write comment
- (void) **error** : (int) number : (char *) name
 | : (int) lineNumber : (int) columnNumber Report error

DConfigTree

Methods

- **init** Init an empty config tree
- **init** : (id src : (char *) name Init config tree with src
- **free** Free config tree
- (BOOL) **read** : (id) srce : (char *) name
 | Read config tree from source
- (BOOL) **write** : (id) dest : (char *) name
 | Write config tree to dest
- (BOOL) **set** : (char *) sect : (char *) opt : (char *) val
 | Set/Insert an option for a section
- (char *) **get** : (char *) sect : (char *) opt . Get an option
- (BOOL) **has** : (char *) sect Check for a section

- (BOOL) **has** : (char *) sect : (char *) opt
 | Check for an option
- (BOOL) **remove** : (char *) sect Remove a section
- (BOOL) **remove** : (char *) sect : (char *) opt
 | Remove an option
- (DList *) **sections** Return a list with all sections
- (DList *) **options** : (char *) section
 | Return all options for a section

DDiscreteDistribution

Methods

- **init** Init discrete distribution
- **free** Free distribution
- (int) **length** Return number values in distribution
- (double) **sum** Return sum values in distribution
- (double) **sumSquared** Return squared sum of values
- (BOOL) **range** : (double) min : (double) max
 | Add score with range [min,max]
- (BOOL) **range** : (int) value
 | Add score with range [v-0.5,v+0.5]
- **reset** Reset length, sums and scores
- (BOOL) **update** : (double) value
 | Update distribution with value
- (double) **mean** Calculate mean of values
- (double) **average** Calculate average of values
- (double) **variance** Calculate variance of values
- (double) **standardDeviation** Calculate SD of values
- (DListIterator *) **scores** Return iterator on score list

DFSM

Methods

- **init** Init empty finite state machine
- **copy** Copy FSM
- **shallowFree** Free FSM, excluding states
- **free** Free FSM, including states
- (BOOL) **isChanged** State change after event?
- (DFSMState *) **current** Get the current state
- (DFSMState *) **previous** Get the previous state
- **transition** : (DFSMState*)o:(DBitArray*)t:(DFSMState*)d
 | Add transition from o to d after event in t
- (DFSMState *) **start** : (DFSMState *) . Start FSM from state
- (DFSMState *) **feed** : (int) event Feed event

DFSMState

Methods

- **init** Init empty FSM state
- **copy** Copy FSM state
- **free** Free FSM state, including transitions

DImageHeader

Constants

- DIM_UNKN Unknown file format
- DIM_JPEG JPEG file format
- DIM_GIF GIF file format
- DIM_PNG PNG file format
- DIM_BMP BMP file format
- DIM_PCX PCX file format
- DIM_IFF IFF file format
- DIM_RAS RAS file format
- DIM_PBM PBM file format
- DIM_PGM PGM file format
- DIM_PPM PPM file format
- DIM_PSD PSD file format
- DIM_SWF SWF file format

Methods

- **init** Init an empty image header
- **init** :(id <DDataReadable>) **file** .. Ini with a data readable
- (BOOL) **inspect** :(id <DDataReadable>) **file**
| Inspect the image via readable
- (char *) **extension** Return file extension for image
- (char *) **mime_type** Return mime type for image
- (DImageHeaderType) **type**... Return the image type DIM_...

DLexer

Methods

- **init** Init lexer
- **init** :(id) **source** :(char *) **name** Init lexer with file
- **free** Free the lexer
- (const char *) **text** Return last scanned text
- (const char *) **name** Return name current file
- (int) **lineNumber** Return current line nr
- (int) **columnNumber** Return current column nr
- (BOOL) **isEof** Check for end of file reached
- **whiteSpace** :(char *) **expr** Set white space expr
- **caseSensitive** :(BOOL) **cs** Set case sensitivity

- (BOOL) **source** :(id) **source** :(char *) **name** Start with source
- (BOOL) **popSource** Pop source
- (BOOL) **nextWhiteSpace** Skip white space
- (BOOL) **nextString** :(char *) **cstring** Scan string
- (BOOL) **nextExpression** :(char *) **cstring** Scan reg exp
- (BOOL) **nextLine** Scan remaining of line
- (BOOL) **checkWhiteSpace** Check white space
- (BOOL) **checkString** :(char *) **cstring** Check string
- (BOOL) **checkExpression** :(char *) **cstring** .. Check reg exp
- **next** Move scanner position after check...
- **next** :(unsigned) **positions** Move scanner position
- **error** :(char *) **msg** Generate error on stderr

DObjcTokenizer

Constants.Generaltokens

- DOT_UNKNOWN Unknown token
- DOT_EOFF End of file token
- DOT_EOFL End of line token
- DOT_WHITESPACE Whitespace token
- DOT_COMMENT Comment token
- DOT_IDENTIFIER Identifier token
- DOT_OPERATOR Operator token
- DOT_PREPROCESSOR Preprocessor token

Constants.Literaltokens

- DOT_DEC_NUMBER Decimal number token
- DOT_OCT_NUMBER Octal number token
- DOT_HEX_NUMBER Hex number token
- DOT_FLP_NUMBER Floating point number token
- DOT_CHAR Character token
- DOT_STRING String token
- DOT_WIDE_CHAR Wide character token
- DOT_WIDE_STRING Wide string token
- DOT_OBJC_STRING Objective-c string token

Constants.Preprocessortokens

- DOT_PASSERT Assert token
- DOT_PDEFINE Define token
- DOT_PELIF Elseif token
- DOT_PELSE Else token
- DOT PENDIF Endif token
- DOT_PERROR Error token
- DOT_PIDENT Ident token
- DOT_PIF If token
- DOT_PIFDEF Ifdef token

- DOT_PIFNDEF Ifndef token
- DOT_PINCLUDE Include token
- DOT_PIMPORT Import token
- DOT_PLINE Line token
- DOT_PPPragma Pragma token
- DOT_PUNASSERT Unassert token
- DOT_PUNDEF Undefine token
- DOT_PWARNING Warning token

Constants.Predefinedtypes

- DOT_TCHAR Char token
- DOT_TDOUBLE Double token
- DOT_TFLOAT Float token
- DOT_TINT Integer token
- DOT_TSHORT Short token
- DOT_TLONG Long token
- DOT_TUNSIGNED Unsigned token

Constants.Storageetokens

- DOT_AUTO Auto token
- DOT_CONST Const token
- DOT_EXTERN Extern token
- DOT_REGISTER Register token
- DOT_STATIC Static token
- DOT_VOLATILE Volatile token

Constants.Typedefinitiontokens

- DOT_ENUM Enum token
- DOT_STRUCT Struct token
- DOT_TYPEDEF Typedef token
- DOT_UNION Union token

Constants.Flowkeywords

- DOT_BREAK Break token
- DOT_CASE Case token
- DOT_CONTINUE Continu token
- DOT_DEFAULT Default token
- DOT_DO Do token
- DOT_ELSE Else token
- DOT_FOR For token
- DOT_GOTO Goto token
- DOT_IF If token
- DOT_RETURN Return token
- DOT_SWITCH Switch token
- DOT_WHILE While token

Constants.Objectiveckeywords

- DOT_INTERFACE Interface token
- DOT_IMPLEMENTATION Implementation token
- DOT_PROTOCOL Protocol token

DOT_END	End token
DOT_PRIVATE	Private token
DOT_PROTECTED	Protected token
DOT_PUBLIC	Public token
DOT_SELECTOR	Selector token
DOT_CLASS	Class
DOT_ENCODE	Encode token
DOT_DEFS	Defs token
DOT_TRY	Try token
DOT_CATCH	Catch token
DOT_FINALLY	Finally token
DOT_THROW	Throw token
DOT_SYNCHRONIZED	Synchronized token
DOT_SYNCHRONIZE	Synchronize token

Constants.Logicaloperators

DOT_AND	And token
DOT_OR	Or token
DOT_NOT	Not token

Constants.Comparisonoperators

```
DOT_SMALLER ..... Smaller token
DOT_SMALLER_EQUAL ..... Smaller equal token
DOT_GREATER ..... Greater token
DOT_GREATER_EQUAL ..... Greater equal token
DOT_EQUAL ..... Equal token
DOT_NOT_EQUAL ..... Not equal token
```

Constants.Bitwiseoperators

<code>DOT_BIT_AND</code>	Bit and token
<code>DOT_BIT_OR</code>	Bit or token
<code>DOT_BIT_XOR</code>	Bit xor token
<code>DOT_BIT_NOT</code>	Bit not token
<code>DOT_BIT_LEFT</code>	Bit left token
<code>DOT_BIT_RIGHT</code>	Bit right token

Constants.Generaloperators

```
DOT_COLON ..... Colon token
DOT_SEMI_COLON ..... Semi-colon token
DOT_COMMA ..... Comma token
DOT_BLOCK_OPEN ..... Block open token
DOT_BLOCK_CLOSE ..... Block close token
DOT_BRACE_OPEN ..... Brace open token
DOT_BRACE_CLOSE ..... Brace close token
DOT_BRACKET_OPEN ..... Bracket open token
DOT_BRACKET_CLOSE ..... Bracket close token
DOT_DEREFERENCE ..... Dereference token
DOT_FIELD ..... Field token
DOT_CONDITION ..... Condition token
```

```

    | Check for a flow keyword token
+ (BOOL) isKeyword :(int) token.. Check for a keyword token
+ (BOOL) isObjcKeyword :(int) token

```

```

| Return objc-keyword description
+ (char *) directive :(int) token
| Return directive description

```

```

- (DObjcTokenizer *) init ..... Init default tokenizer
- init :(id) source :(char *) name Init tokenizer with source
- free ..... Free the tokenizer
- (char *) text ..... Return the last scanned text
- (char *) name ..... Return the name of the source
- (int) lineNumber ..... Return the current line number
- (int) columnNumber ..... Return the current column number
- (BOOL) isEOF ..... Check if the end of the source is reached
- (BOOL) skipWhiteSpace Check if white space must be skipped
- skipWhiteSpace :(BOOL) skip ..... Set skip white space
- (BOOL) source :(id) source :(char *) name
    | Start tokenizing source
- (BOOL) popSource ..... Pop source for sources stack
- (int) nextToken ..... Scan source for next token
- (int) checkToken ..... Check source for next token
- next ..... Move source location after checkToken
- error :(char *) msg ..... Generate error on stderr

```

DProperty

```

- init ..... Init empty property
- init :(char *) name :(id) value ..... Init property
- init :(char *) name ..... Init property group
- deepen ..... Deepen a property after copy
- free ..... Free a property
- (BOOL) isGroup ..... Check if property is a group
- (const char *) name ..... Return name of property
- (id) value ..... Return value reference of the property
- property :(char *) name :(id) value ..... Set the property
- group :(char *) name ..... Set the property group

```

DPropertyTree

Methods

- **init** Init default property tree
- **init** : (char *) **name** Init named property tree
- **free** Free the property tree
- (char *) **name** Return the name of the tree
- **name** : (char *) **name** Set the name of the tree
- **group** : (DProperty *) **parent** : (char *) **name**
 - | Add group property to the property tree
- **property** : (DProperty *) **parent** : (char *) **name**
 - | : (id) **value** Add property to the property tree
- (BOOL) **read** : (id) **source** : (char *) **name**
 - | Read the property tree and set the value references
- (BOOL) **write** : (id) **destination** : (char *) **name**
 - | Write the property tree by reading the value references
- (BOOL) **remove** : (DProperty *) **property** .. Remove property
- (BOOL) **remove** : (DProperty *) **parent** : (char *) **name**
 - | Remove property by name

DSource

Methods

- **init** Init empty source
- **init** : (id) **source** : (char *) **name** Init source with file
- **free** Free the source
- (const char *) **name** Return the name current source
- (const char *) **line** Return remaining line
- (int) **lineNumber** Return current line number
- (int) **columnNumber** Return the current column number
- (BOOL) **set** : (id) **source** : (char *) **name** . Start with source
- (BOOL) **nextLine** Get the next line
- (BOOL) **appendLine** Append the next line
- (BOOL) **scanned** : (unsigned) **lengthSet** length chars scanned
- (BOOL) **isLineScanned** Check if full line scanned
- (DSource *) **error** : (char *) **msg** .. Generate error on stderr

DTokenizer

Methods

- **init** Init tokenizer
- **init** : (id) **source** : (char *) **name** ... Init tokenizer with file
- **free** Free the tokenizer
- (char *) **text** Return last scanned text
- (const char *) **name** Return name current file
- (int) **lineNumber** Return current line number

- (int) **columnNumber** Return current column number
- (BOOL) **isEof** Check for end of file
- **skipWhiteSpace** : (BOOL) **skip** Set skip white space
- (BOOL) **skipWhiteSpace** Return skip white space
- (BOOL) **source** : (id) **source** : (char *) **name** Start with source
- (BOOL) **popSource** Pop source from sources stack
- (int) **nextToken** Scan source for token
- (int) **checkToken** Check source for token
- **next** Move source location after checkToken
- **next** : (unsigned) **positions** Move source location
- **error** : (char *) **msg** Generate error on stderr

Basicscanners to be overridden

- (DText *) **whiteSpace** : (char *) **text** White space scanner
- (DText *) **comment** : (char *) **text** Comment scanner
- (DText *) **keyword** : (char *) **text** Keyword scanner
- (DText *) **string** : (char *) **text** String scanner
- (DText *) **operator** : (char *) **text** Operator scanner
- (DText *) **number** : (char *) **text** Number scanner
- (DText *) **extraToken1** : (char *) **text** Extra scanner
- (DText *) **extraToken2** : (char *) **text** Extra scanner
- (DText *) **extraToken3** : (char *) **text** Extra scanner

DXMLElement

Constants

- DXML_ELEMENT XML Element
- DXML_ATTRIBUTE XML Attribute
- DXML_TEXT XML Text
- DXML_CDATA XML CDATA Text
- DXML_PI XML Processing Instruction
- DXML_COMMENT XML Comment
- DXML_DOCUMENT XML Document
- DXML_NAMESPACE XML Namespace

Methods

- **init** Init empty xml node
- **init** : (int) **type** : (char *) **name** : (char *) **value**
 - | Init xml node with type, name and value
- **deepen** Deepen the copied node
- **free** Free the node
- (int) **type** Return the type of the node
- (char *) **name** Return the name of the node
- (char *) **value** Return the value of the node
- **set** : (int) **type** : (char *) **name** : (char *) **value**
 - | Set the type, name and value of the node

DXMLElement : DTree

Methods

- (DXMLElement *) **init** Init empty xml tree
- **init** : (id) **source** : (char *) **name** : (char) **separator**
 - | Init xml tree with xml source
- **free** Free the tree and the stored nodes
- **shallowFree** Free the tree, not the stored nodes
- (BOOL) **read** : (id) **source** : (char *) **name** : (char) **sep**
 - | Build xml tree from xml source
- (BOOL) **write** : (id) **destination** : (char *) **name**
 - | Write xml tree to destination

DXMLElementWriter

Methods

- **init** Init default xml writer
- **init** : (id) **destination** : (char) **separator**
 - | Init xml writer with a destination
- **free** Free the writer
- (BOOL) **start** : (id) **destination** : (char) **separator**
 - | Start the writing of a xml file
- (BOOL) **startDocument** : (char *) **version** : (char *) **enc**
 - | : (int) **standalone** Write start of document
- (BOOL) **endDocument** Write end of document
- (BOOL) **startElement** : (char *) **name** . Write start of element
- (BOOL) **attribute** : (char *) **attr** : (char *) **value**
 - | Write attribute
- (BOOL) **endElement** Write end of element
- (BOOL) **characters** : (char *) **text** Write text
- (BOOL) **comment** : (char *) **text** Write comment
- (BOOL) **processingInstruction** : (char *) **target**
 - | : (char *) **value** Write PI
- (BOOL) **startCDATA** Write start of CDATA section
- (BOOL) **endCDATA** Write end of CDATA section
- (BOOL) **startNamespace** : (char *) **prefix** : (char *) **uri**
 - | Write start of namespace
- (BOOL) **endNamespace** Write end of namespace
- (BOOL) **unparsed** : (char *) **text** Write unparsed text
- (void) **error** : (int) **number** : (const char *) **name**
 - | : (int) **lineNumber** : (int) **columnNumber** Report error

Version 0.7.0. This card may be freely distributed under the terms of the GNU general public licence

Copyright © 2003-2005 by Dick van Oudheusden