Trucler A Common Lisp environment protocol and its implementation.

Robert Strandh

2019

Contents

1	Intr	oducti	on	1
2	Que	erying	the environment	3
	2.1	Query	functions	4
		2.1.1	Variable information	4
		2.1.2	Function information	4
		2.1.3	Block information	5
		2.1.4	Tag information	5
		2.1.5	Class information	6
		2.1.6	Optimize information	6
	2.2	Mixin	classes	6
		2.2.1	name-mixin	7
		2.2.2	identity-mixin	7
		2.2.3	type-mixin	7
		2.2.4	ignore-mixin	8
		2.2.5	dynamic-extent-mixin	8
		2.2.6	expansion-mixin	8
		2.2.7	expander-mixin	9
		2.2.8	class-name-mixin	9
		2.2.9	inline-mixin	9
		2.2.10	method-class-name-mixin	10
		2.2.11	speed-mixin	10
		2.2.12	compilation-speed-mixin	11
		2.2.13	debug-mixin	11
		2.2.14	space-mixin	11
		2.2.15	safety-mixin	12
		2.2.16	superclass-names-mixin	12

		2.2.17	metaclass-name-mixin	13
	2.3	Abstra	act query classes	13
	2.4	Instan	tiable classes	14
		2.4.1	Variable description	14
		2.4.2	Function description	15
		2.4.3	Block description	16
		2.4.4	Tag description	17
		2.4.5	Class description	17
		2.4.6	Optimize description	17
3	Aug	gmentii	ng the environment	19
	3.1^{-1}	Creati	ng new description	19
	3.2	Low-le	vel augmentation functions	19
	3.3	Mergir	ng descriptions	21
	3.4	High-le	evel annotation functions	26
		3.4.1	Adding and annotating variables	26
		3.4.2	Adding and annotating functions	29
		3.4.3	Adding blocks	31
		3.4.4	Adding tags	32
		3.4.5	Annotating the optimize qualities	32
Bi	bliog	raphy		35
In	\mathbf{dex}			36

Chapter 1

Introduction

In section 8.5 of the second edition of the book "Common Lisp, the Language" (also known as CLtL2) by Guy Steele [Ste90], a protocol for accessing compiletime environments is defined. That protocol has two main problems:

- 1. It is incomplete. It does not provide for a way to query the environment for description about blocks or tags.
- 2. It is not extensible. In order for an implementation to make one of the query functions return more information, additional return values would have to be defined. However, such a change is not considered backward compatible, so this kind of extension is not recommended.

Trucler introduces a protocol that solves these problems as follows:

- 1. It contains additional query and augmentation functions for blocks and tags.
- 2. Instead of returning multiple values, the query functions return standard objects. Accessors specialized to the classes of those objects provide the information that the protocol in CLtL2 provides as multiple values.

In addition to providing a mechanism that solves the problems of protocol

presented in CLtL2, we also add several new features that a language processor must use to process source code.

Chapter 2

Querying the environment

In this chapter, we describe classes and functions that are used by the language processor to query the environment concerning information about program elements that the language processor needs in order to determine how to process those program elements.

When the language processor calls a generic query function, it passes two or three arguments, depending on the function it calls. The first argument is called the client. Trucker does not specialize on this argument, but client code should define a standard class and pass an instance of that class for this argument. Client code can then define auxiliary methods that specialize to this class on the query generic functions. The second argument is the environment concerned by the query. Client code must supply methods on these functions, specialized to its particular representation of its global environments. If the client does not have an explicit representation of its global environment (as is usually the case), it must nevertheless define a dummy standard class to specialize on. Contrary to global environments, Trucler provides its own representation of *lexical* environments, and it provides methods on the query functions, specialized to the classes defined to represent those lexical environments. Client code that wants to use a different representation of lexical environments than the one provided by Trucler must also provide methods specialized to its lexical environment classes.

The primary methods on the query functions should return instances of the

classes described in this chapter. Any such instance contains all available information about some program element in that particular environment. This information must typically be assembled from different parts of the environment. For that reason, client code typically creates a new instance whenever a query function is called, rather than attempting to store such instances in the environment. If any of these client-supplied methods fails to accomplish its task, it should return nil.

Client code is free to define subclasses of the classes described here, for instance in order to represent implementation-specific information about the program elements. Client code would then typically also provide auxiliary methods or overriding primary methods on the compilation functions that handle these classes.

2.1 Query functions

2.1.1 Variable information

⇒	describe-variable	client	environment	name	
---	-------------------	--------	-------------	------	--

This function is called by the language processor whenever a symbol in a *variable* position is to be compiled. If successful, it returns an instance of one of the classes described in Section 2.4.1.

 \Rightarrow no-variable-description

This condition is signaled by Trucler when a client-supplied method on the generic function describe-variable returns nil.

 \Rightarrow name (condition no-variable-description)

This method returns the name of the variable for which no description was available.

2.1.2 Function information

 \Rightarrow describe-function client environment name [Generic Function]

This function is called by the language processor whenever a symbol in a func-

[Generic Function]

[Condition]

[Method]

tion position is to be compiled or whenever a function name is found in a context where it is known to refer to a function. If successful, it returns an instance of one of the classes described in Section 2.4.2.

 \Rightarrow no-function-description

This condition is signaled by Trucler when a client-supplied method on the generic function describe-function returns nil.

 \Rightarrow name (condition no-function-description)

This method returns the name of the function for which no description was available.

2.1.3 Block information

 \Rightarrow describe-block client environment name

This function is called by the language processor whenever a symbol referring to a *block* is found, typically in a **return-from** form. If successful, it returns an instance of the class described in Section 2.4.3.

```
\Rightarrow no-block-description
```

This condition is signaled by Trucler when a client-supplied method on the generic function describe-block returns nil.

 \Rightarrow name (condition no-block-description)

This method returns the name of the block for which no description was available.

2.1.4 Tag information

 \Rightarrow describe-tag client environment tag

This function is called by the language processor whenever a symbol or an integer referring to a tag is found, typically in a go form. If successful, it returns an instance of the class described in Section 2.4.4.

 \Rightarrow no-tag-description

This condition is signaled by Trucler when a client-supplied method on the

[Condition]

[Method]

[Method]

[Condition]

[Generic Function]

[Generic Function]

[Condition]

generic function describe-tag returns nil.

 \Rightarrow name (condition no-tag-description)

This method returns the name of the tag for which no description was available.

2.1.5 Class information

 \Rightarrow describe-class *client environment class-name*

This function is called by the language processor whenever a symbol referring to a *class* is found, for example as a specializer in a **defmethod** form. If successful, it returns an instance of the class described in Section 2.4.5.

 \Rightarrow no-class-description

This condition is signaled by Trucler when a client-supplied method on the generic function describe-class returns nil.

```
\Rightarrow name (condition no-class-description)
```

This method returns the name of the class for which no description was available.

2.1.6 Optimize information

 \Rightarrow describe-optimize *client* environment

Client-supplied methods on this function must always return a valid instance of the class optimize-description. It returns an instance of the class described in Section 2.4.6.

2.2 Mixin classes

For maximum flexibility, each query class is the subclass of one or more mixin classes, each one providing one single feature. That feature is represented as a slot with an initarg, a reader, an initform, and a type.

[Condition]

[Generic Function]

[Method]

[Method]

[Generic Function]

2.2.1 name-mixin

name-mixin

 \Rightarrow

This class is a superclass of query classes that require a name to identify the information supplied by the class instances.

 $\Rightarrow :name \qquad [Initarg]$ $\Rightarrow name (description name-mixin) \qquad [Method]$

Given an instance of the class **name-mixin**, this method returns the name information, as supplied by the initarg :**name**.

2.2.2 identity-mixin

 \Rightarrow identity-mixin

This class is a superclass of query classes that require some kind of identity to distinguish instances of the query class that have the same name.

\Rightarrow	:identity	[Initarg]
	•	

 \Rightarrow identity (description identity-mixin) [Method]

Given an instance of the class identity-mixin, this method returns the identity information, as supplied by the initarg :idenity.

2.2.3 type-mixin

 \Rightarrow type-mixin

This class is a superclass of query classes that provide information about entities that can have a type.

\Rightarrow :type	[Initarg]

If this initarg is not supplied, it defaults to t.

 \Rightarrow type (description type-mixin)

Given an instance of the class type-mixin, this method returns the type information, as supplied by the initarg :type.

[Class]

[Class]

[Class]

[Method]

2.2.4 ignore-mixin

This class is a superclass of query classes that provide information about entities that can be declared **ignore** or **ignorable**.

The value of this initiang must be one of the symbols ignore and ignorable from the common-lisp package.

Given an instance of the class **ignore-mixin**, this method returns the ignore information, as supplied by the initarg **:ignore**.

2.2.5 dynamic-extent-mixin

dynamic-extent-mixin

ignore (description ignore-mixin)

This class is a superclass of query classes that provide information about entities that can be declared dynamic-extent.

\Rightarrow	:dynamic-extent		
\Rightarrow	dynamic-extent	(description dynamic-extent-mixin)) [Method]

Given an instance of the class dynamic-extent-mixin, this method returns the dynamic-extent information, as supplied by the initarg :dynamic-extent.

2.2.6 expansion-mixin

```
\Rightarrow expansion-mixin
```

This class is a superclass of query classes that provide information about entities that can have an expansion. In particular, it is a superclass of the abstract class symbol-macro-description.

\Rightarrow	:expansion	[Initarg]
\Rightarrow	expansion $(description expansion-mixin)$	[Method]

8

ignore-mixin

:ignore

 \Rightarrow

 \Rightarrow

 \Rightarrow

 \Rightarrow

[Initarg]

[Method]

[Class]

[Class]

2.2. MIXIN CLASSES

Given an instance of the class expansion-mixin, this method returns the expansion information, as supplied by the initiarg :expansion.

2.2.7 expander-mixin

 \Rightarrow expander-mixin

 \Rightarrow :expander

This class is a superclass of query classes that provide information about entities that can have an expander function. In particular, it is a superclass of the abstract class macro-description.

	=		
\Rightarrow	expander	(description expander-mixin)	[Method]

Given an instance of the class expander-mixin, this method returns the expander information, as supplied by the initarg :expander.

2.2.8 class-name-mixin

\Rightarrow class-name-mixin

This class is a superclass of query classes that provide information about entities that can have a class-name. In particular, it is a superclass of the class global-function-description.

\Rightarrow	:class-name	[Initarg]
⇒	class-name (description class-name-mixin)	[Method]

Given an instance of the class class-name-mixin, this method returns the class-name information, as supplied by the initiarg :class-name.

2.2.9 inline-mixin

 \Rightarrow inline-mixin

This class is a superclass of query classes that provide information about entities that can have inline information. In particular, it is a superclass of the class authentic-function-description.

[Class]

[Init arg]

[Class]

Possible values for this initiarg are nil, inline, and notinline, all symbols in the common-lisp package. The value nil means that no inline information has been provided, and this is the default value if the initiar is omitted.

Given an instance of the class inline-mixin, this method returns the inline information, as supplied by the initarg :inline.

2.2.10method-class-name-mixin

method-class-name-mixin \Rightarrow

 \Rightarrow inline (description inline-mixin)

This class is a superclass of query classes that provide information about entities that can have method-class-name information. In particular, it is a superclass of the class generic-function-description.

:method-class-name \Rightarrow

> The value of this initiating is a symbol naming a class to be used for methods. If this initiarg is not given, it defaults to the symbol standard-method.

\Rightarrow	method-class-name	$(description \ \texttt{method-class-name-mixin})$) [Method]
---------------	-------------------	--	------------

Given an instance of the class method-class-name-mixin, this method returns the method-class-name information, as supplied by the initarg :method-class-name.

2.2.11speed-mixin

```
This class is a superclass of query classes that provide information about entities
that can have speed information. In particular, it is a superclass of the class
optimize-description.
```

 \Rightarrow :speed

 \Rightarrow

The value of this initiarg must be an integer between 0 and 3 inclusive.

speed (description speed-mixin) \Rightarrow

:inline \Rightarrow

speed-mixin

[Initarg]

[Method]

[Class]

[Initarg]

[Method]

[Class]

[Initarg]

Given an instance of the class **speed-mixin**, this method returns the compilationspeed information, as supplied by the initiarg :speed.

2.2.12compilation-speed-mixin

compilation-speed-mixin \Rightarrow

> This class is a superclass of query classes that provide information about entities that can have compilation-speed information. In particular, it is a superclass of the class optimize-description.

\Rightarrow :compilation-speed

The value of this initiarg must be an integer between 0 and 3 inclusive.

compilation-speed (description compilation-speed-mixin) [Method] \Rightarrow

Given an instance of the class compilation-speed-mixin, this method returns the compilation-speed information, as supplied by the initarg :compilation-speed.

2.2.13debug-mixin

debug-mixin \Rightarrow

> This class is a superclass of query classes that provide information about entities that can have debug information. In particular, it is a superclass of the class optimize-description.

:debug \Rightarrow

The value of this initiarg must be an integer between 0 and 3 inclusive.

 \Rightarrow debug (description debug-mixin)

Given an instance of the class debug-mixin, this method returns the debug information, as supplied by the initarg :debug.

2.2.14space-mixin

space-mixin \Rightarrow

[Class]

[Initarg]

[Method]

[Class]

[Initarg]

This class is a superclass of query classes that provide information about entities that can have space information. In particular, it is a superclass of the class optimize-description.

The value of this initiarg must be an integer between 0 and 3 inclusive.

 \Rightarrow space (description space-mixin)

Given an instance of the class **space-mixin**, this method returns the space information, as supplied by the initiarg **:space**.

2.2.15 safety-mixin

 \Rightarrow safety-mixin

This class is a superclass of query classes that provide information about entities that can have safety information. In particular, it is a superclass of the class optimize-description.

 \Rightarrow :safety

The value of this initiang must be an integer between 0 and 3 inclusive.

 \Rightarrow safety (description safety-mixin)

Given an instance of the class **safety-mixin**, this method returns the safety information, as supplied by the initarg **:safety**.

2.2.16 superclass-names-mixin

```
\Rightarrow superclass-names-mixin
```

This class is a superclass of query classes that provide information about entities that can have superclass-names information. In particular, it is a superclass of the class class-description.

 \Rightarrow :superclass-names

The value of this initarg is a list of symbols naming a classes. If this initarg is not given, it defaults the empty list. Only explicitly mentioned superclass names should be provided.

:space

 \Rightarrow

[Initarg]

[Method]

[Initarg]

[Class]

[Method]

[Class]

Initarg

2.3. ABSTRACT QUERY CLASSES

 \Rightarrow superclass-names (description superclass-names-mixin) [Method]

Given an instance of the class superclass-names-mixin, this method returns the superclass-names information, as supplied by the initiarg :superclass-names.

2.2.17 metaclass-name-mixin

```
\Rightarrow metaclass-name-mixin
```

This class is a superclass of query classes that provide information about entities that can have metaclass-name information. In particular, it is a superclass of the class class-description.

 \Rightarrow :metaclass-name

The value of this initarg is a symbol naming a class to be used as a metaclasss. If this initarg is not given, it defaults to the symbol standard-class.

\Rightarrow metaclass-name	(description metaclass-name-mixin)	Method
------------------------------	-----------------------------------	---	--------

Given an instance of the class metaclass-name-mixin, this method returns the metaclass-name information, as supplied by the initarg :metaclass-name.

2.3 Abstract query classes

```
\Rightarrow variable-description
```

This abstract class is the superclass of every query class returned by a call to the generic function describe-variable. It is a subclass of the class name-mixin.

 \Rightarrow authentic-variable-description

This abstract class is a subclass of the classes variable-description and type-mixin.

It is a superclass of the two instantiable classes lexical-variable-description and special-variable-description.

```
\Rightarrow symbol-macro-description
```

This abstract class is a subclass of the classes variable-description, type-mixin, and expansion-mixin.

[Class]

[Initarg]

[Class]

[Class]

It is a superclass of the two instantiable classes local-symbol-macro-description and global-symbol-macro-description.

 \Rightarrow function-description

This abstract class is the superclass of every query class returned by a call to the generic function function-description. It is a subclass of the class name-mixin.

 \Rightarrow authentic-function-description

This abstract class is a subclass of the classes function-description and type-mixin.

It is a superclass of the two instantiable classes local-function-description and global-function-description.

 \Rightarrow macro-description

This abstract class is a subclass of the classes function-description and expander-mixin.

It is a superclass of the two instantiable classes local-macro-description and global-macro-description.

2.4 Instantiable classes

2.4.1 Variable description

 \Rightarrow lexical-variable-description

This class represents information about lexical variables. An instance of this class is returned by a call to variable-description when it turns out that the symbol passed as an argument refers to a lexical variable.

This class is a subclass of the classes authentic-variable-description identity-mixin, ignore-mixin, and dynamic-extent-mixin.

 \Rightarrow special-variable-description

This class represents information about special variables. An instance of this class is returned by a call to variable-description when it turns out that

[Class]

[Class]

[Class]

[Class]

[Class]

14

2.4. INSTANTIABLE CLASSES

the symbol passed as an argument refers to a special variable.

This class is a subclass of the classes authentic-variable-description and global-p-mixin.

 \Rightarrow constant-variable-description

This class represents information about constant variables. An instance of this class is returned by a call to variable-description when it turns out that the symbol passed as an argument refers to a constant variable.

This class is a subclass of the classes variable-description and value-mixin.

```
\Rightarrow global-symbol-macro-description
```

This class is a subclass of symbol-macro-description. It is returned by a call to variable-descriptionrmation when the name is defined as a global symbol macro, as defined by define-symbol-macro.

 \Rightarrow local-symbol-macro-description

This class is a subclass of symbol-macro-description and ignore-mixin. It is returned by a call to variable-descriptionrmation when the name is defined as a local symbol macro, as defined by symbol-macrolet.

2.4.2 Function description

 \Rightarrow local-function-description

This class represents information about local functions introduced by flet or labels. An instance of this class is returned by a call to function-description when it turns out that the function name passed as an argument refers to a local function.

This class is a subclass of authentic-function-description, identity-mixin, ignore-mixin, and dynamic-extent-mixin.

 \Rightarrow global-function-description

This class represents information about global functions. An instance of this class is returned by a call to function-description when it turns out that the function name passed as an argument refers to a global function.

[Class]

[Class]

[Class]

[Class]

This class is a subclass of authentic-function-description, compiler-macro-mixin, and class-name-mixin.

 \Rightarrow generic-function-description

This class is a subclass of global-function-description and method-class-name-mixin.

 \Rightarrow local-macro-description

This class represents information about local macros introduced by macrolet. An instance of this class is returned by a call to function-description when it turns out that the function name passed as an argument refers to a local macro.

This class is a subclass of macro-description and ignore-mixin.

 \Rightarrow global-macro-description

This class represents information about global macros introduced by macrolet. An instance of this class is returned by a call to function-description when it turns out that the function name passed as an argument refers to a global macro.

This class is a subclass of macro-description and compiler-macro-mixin.

 \Rightarrow special-operator-description

This class represents information about special operators. An instance of this class is returned by a call to function-description when it turns out that the function name passed as an argument refers to a special operator.

This class is a subclass of the class function-description.

2.4.3**Block** description

 \Rightarrow block-description

This class represents information about blocks introduced by block. An instance of this class is returned by a call to **block-description** when the symbol passed as an argument refers to a block.

This class is a subclass of the classes name-mixin and identity-mixin.

16

[Class]

[Class]

[Class]

[Class]

2.4.4 Tag description

\Rightarrow tag-description

This class represents information about tags introduced by tagbody. An instance of this class is returned by a call to tag-description when the name (which must be a symbol or an integer) passed as an argument refers to a tag.

This class is a subclass of the classes name-mixin and identity-mixin.

2.4.5 Class description

\Rightarrow class-description

This class represents information about a class introduced by defclass. An instance of this class is returned by a call to class-description when the name (which must be a symbol) passed as an argument refers to a class.

This class is a subclass of the classes name-mixin, superclass-names-mixin, and metaclass-name-mixin.

2.4.6 Optimize description

 \Rightarrow optimize-description

This class is a subclass of speed-mixin, compilation-speed-mixin, debug-mixin, space-mixin, and safety-mixin.

[Class]

[Class]

Chapter 3

Augmenting the environment

3.1 Creating new description

In order to create a new description, make-instance must be called, providing values for all the initialization arguments corresponding to features that do not have any initialization forms.

3.2 Low-level augmentation functions

In this section, we describe basic functions for augmenting an environment, given an old environment and a description.

 \Rightarrow augment-with-variable-description client environment description [Generic Function]

This function is used to create a new environment object from an existing environment object and an instance of the class variable-description.

```
⇒ augment-with-variable-description
client
(environment environment)
(description variable-description) [Method]
```

This default method returns a new environment object which is like the one passed as an argument, except that *description* will shadow any variable de-

scription with the same name.

\Rightarrow augment-with-function-description client environment function-description [Generic Function]

This function is used to create a new environment object from an existing environment object and an instance of the class function-description.

```
⇒ augment-with-function-description
client
(environment environment)
(function-description function-description) [Method]
```

This default method returns a new environment object which is like the one passed as an argument, except that *function-description* will shadow any function description with the same name.

 \Rightarrow augment-with-block-description client environment block-description [Generic Function]

This function is used to create a new environment object from an existing environment object and an instance of the class **block-description**.

```
⇒ augment-with-block-description
client
(environment environment)
(block-description block-description) [Method]
```

This default method returns a new environment object which is like the one passed as an argument, except that *block-description* will shadow any block description with the same name.

```
\Rightarrow augment-with-tag-description client environment tag-description [Generic Function]
```

This function is used to create a new environment object from an existing environment object and an instance of the class tag-description.

```
⇒ augment-with-tag-description
client
(environment environment)
(tag-description tag-description) [Method]
```

This default method returns a new environment object which is like the one passed as an argument, except that *tag-description* will shadow any tag description with the same name.

3.3. MERGING DESCRIPTIONS

augment-with-optimize-description ⇒ client environment optimize description [Generic Function]

This function is used to create a new environment object from an existing environment object and an instance of the class optimize-description.

```
\Rightarrow augment-with-optimize-description
    client
    (environment environment)
    (optimize-description optimize-description)
                                                                                      [Method]
```

This default method returns a new environment object which is like the one passed as an argument, except that *optimize-description* will shadow any previous optimize description.

3.3Merging descriptions

We use the term *merging* to mean the creation of a new description from an existing description plus some additional information such as type or dynamic extent.

In this section, we describe generic functions that are provided for this purpose.

```
\Rightarrow merge-type client description type
```

Given an instance of the class description and a type descriptor, return a new instance that is just like *description* (including the class and the values of all the slots), except that its type description has been updated according to that of type.

```
\Rightarrow invalid-description-for-merging-type-information
                                                                                   [Condition]
```

This condition is signaled by merge-type when the *description* argument is not an instance of a class that contains information about type.

```
\Rightarrow merge-type client description type
```

This is the default method provided on merge-type. It signals the condition invalid-description-for-merging-type-information.

```
\Rightarrow merge-type
     client
     (description type-mixin)
```

21

[Generic Function]

```
[Method]
```

type

This is the main method provided on merge-type and it is specialized to type-mixin.

 \Rightarrow merge-ignore client description ignore

Given an instance of the class description and one of the symbols cl:ignore and cl:ignorable, return a new instance that is just like *description* (including the class and the values of all the slots), except that its ignore information has been updated according to that of *ignore*.

\Rightarrow	invalid-description-for-merging-ignore-information	[Condition]
---------------	--	-------------

This condition is signaled by merge-ignore when the *description* argument is not an instance of a class that contains information about ignore.

 \Rightarrow merge-ignore client description ignore

This is the default method provided on merge-ignore. It signals the condition invalid-description-for-merging-ignore-information.

```
\Rightarrow merge-ignore
    client
     (description ignore-mixin)
    ignore
```

This is the main method provided on merge-ignore and it is specialized to ignore-mixin.

 \Rightarrow merge-dynamic-extent *client description*

Given an instance of the class description, return a new instance that is just like *description* (including the class and the values of all the slots), except that its dynamic-extent information has been updated so that it is *true*.

 \Rightarrow invalid-description-for-merging-dynamic-extent-information [Condition]

This condition is signaled by merge-dynamic-extent when the *description* argument is not an instance of a class that contains information about dynamicextent.

 \Rightarrow merge-dynamic-extent *client description*

This is the default method provided on merge-dynamic-extent. It signals the condition invalid-description-for-merging-dynamic-extent-information.

[Method]

[Generic Function]

[Method]

[Method]

[Method]

[Generic Function]

22

3.3. MERGING DESCRIPTIONS

 \Rightarrow merge-dynamic-extent *client* (description dynamic-extent-mixin) [Method]

This is the main method provided on merge-dynamic-extent and it is specialized to dynamic-extent-mixin.

 \Rightarrow merge-inline client description inline [Generic Function]

Given an instance of the class description and one of the symbols cl:inline and cl:notinline, return a new instance that is just like *description* (including the class and the values of all the slots), except that its inline information has been updated according to that of *inline*.

\Rightarrow invalid-description-for-merging-inline-information [Condition]

This condition is signaled by merge-inline when the *description* argument is not an instance of a class that contains information about inline.

 \Rightarrow merge-inline client description inline

This is the default method provided on merge-inline. It signals the condition invalid-description-for-merging-inline-information.

⇒ merge-inline client (description inline-mixin) inline

This is the main method provided on **merge-inline** and it is specialized to inline-mixin.

 \Rightarrow merge-speed *client description value*

Given an instance of the class description and an integer between 0 and 3, return a new instance that is just like *description* (including the class and the values of all the slots), except that its speed information has been updated according to that of *value*.

 \Rightarrow invalid-description-for-merging-speed-information [Condition]

This condition is signaled by **merge-speed** when the *description* argument is not an instance of a class that contains information about speed.

 \Rightarrow merge-speed client description speed

This is the default method provided on merge-speed. It signals the condition invalid-description-for-merging-speed-information.

[Method]

[Method]

[Method]

[Generic Function]

23

```
⇒ merge-speed
client
(description speed-mixin)
value
```

[Method]

[Generic Function]

This is the main method provided on merge-speed and it is specialized to speed-mixin.

 \Rightarrow merge-compilation-speed *client description value*

Given an instance of the class description and an integer between 0 and 3, return a new instance that is just like *description* (including the class and the values of all the slots), except that its compilation-speed information has been updated according to that of *value*.

 \Rightarrow invalid-description-for-merging-compilation-speed-information [Condition]

This condition is signaled by merge-compilation-speed when the *description* argument is not an instance of a class that contains information about compilation-speed.

```
\Rightarrow merge-compilation-speed client description compilation-speed [Method]
```

This is the default method provided on merge-compilation-speed. It signals the invalid-description-for-merging-compilation-speed-information condition.

```
\Rightarrow merge-compilation-speed
```

client
(description compilation-speed-mixin)
value

[Method]

This is the main method provided on merge-compilation-speed and it is specialized to compilation-speed-mixin.

 \Rightarrow merge-debug client description value

[Generic Function]

Given an instance of the class description and an integer between 0 and 3, return a new instance that is just like *description* (including the class and the values of all the slots), except that its debug information has been updated according to that of *value*.

\Rightarrow	invalid-description-for-merging-debug-information	[Condition]

This condition is signaled by merge-debug when the *description* argument is not an instance of a class that contains information about debug.

3.3. MERGING DESCRIPTIONS

 \Rightarrow merge-debug client description debug

This is the default method provided on merge-debug. It signals the condition invalid-description-for-merging-debug-information.

 \Rightarrow merge-debug client

(description debug-mixin) value

This is the main method provided on merge-debug and it is specialized to debug-mixin.

 \Rightarrow merge-space client description value

Given an instance of the class description and an integer between 0 and 3, return a new instance that is just like *description* (including the class and the values of all the slots), except that its space information has been updated according to that of value.

[Condition] \Rightarrow invalid-description-for-merging-space-information

This condition is signaled by merge-space when the *description* argument is not an instance of a class that contains information about space.

 \Rightarrow merge-space client description space

This is the default method provided on merge-space. It signals the condition invalid-description-for-merging-space-information.

```
\Rightarrow merge-space
```

client(description space-mixin) value

This is the main method provided on merge-space and it is specialized to space-mixin.

 \Rightarrow merge-safety client description value

Given an instance of the class description and an integer between 0 and 3, return a new instance that is just like *description* (including the class and the values of all the slots), except that its safety information has been updated according to that of *value*.

 \Rightarrow invalid-description-for-merging-safety-information [Condition]

```
[Method]
```

[Method]

[Generic Function]

[Method]

[Method]

[Generic Function]

25

This condition is signaled by merge-safety when the *description* argument is not an instance of a class that contains information about safety.

 \Rightarrow merge-safety client description safety

This is the default method provided on merge-safety. It signals the condition invalid-description-for-merging-safety-information.

```
⇒ merge-safety
client
(description safety-mixin)
value [Method]
```

This is the main method provided on **merge-safety** and it is specialized to **safety-mixin**.

3.4 High-level annotation functions

3.4.1 Adding and annotating variables

Adding a lexical variable

 \Rightarrow add-lexical-variable client environment name &optional identity [Generic Function]

This function returns a new environment that is like *environment* except that it has been augumented with a lexical variable named *name*. The optional argument *identity* can be supplied by client code to distinguish different lexical variables with the same name.

 \Rightarrow add-lexical-variable client (environment environment) name & optional identity [Method]

This is the main method on add-lexical-variable. It instantiates the class lexical-variable-description and then creates a new environment by calling the function augment-with-variable-description.

Adding a special variable

 \Rightarrow add-special-variable *client environment name*

[Generic Function]

[Method]

This function returns a new environment that is like *environment* except that it has been augumented with a special variable named *name*.

$$\Rightarrow$$
 add-special-variable client (environment environment) name [Method]

This is the main method on add-special-variable. It instantiates the class special-variable-description and then creates a new environment by calling the function augment-with-variable-description.

Adding a local symbol macro

 \Rightarrow add-local-symbol-macro client environment name expansion [Generic Function]

This function returns a new environment that is like *environment* except that it has been augmented with a local symbol macro named **name**, with the expansion *expansion*

 \Rightarrow add-local-symbol-macro client (environment environment) name expansion [Method]

This is the main method on add-local-symbol-macro. It instantiates the class local-symbol-macro-description and then creates a new environment by calling the function augment-with-variable-description.

Annotating a variable with a type

 \Rightarrow add-variable-type client environment name type [Generic Function]

This function returns a new environment that is like *environment* except that the variable named *name* has been annotated with the type specifier *type*.

The type of the variable returned when the new environment is queried for the variable named *name* will have a new type that is the conjunction of *type* and the type it had in *environment*.

This function can be used when *name* names a lexical variable, a special variable, or a symbol macro, but *not* when *name* names a constant variable.

 \Rightarrow add-variable-type client (environment environment) name type [Method]

This is the main method on add-variable-type. It calls describe-variable to obtain an existing variable description. It then calls merge-type to create a

new variable description. Finally, it calls augment-with-variable-description in order to create and return a new environment.

Annotating a variable with an ignore declaration

 \Rightarrow add-variable-ignore client environment name ignore [Generic Function]

This function returns a new environment that is like *environment* except that the variable named *name* has been annotated with an **ignore** declaration.

The argument *ignore* must be the symbol **ignore** or the symbol **ignorable**.

This function can be used when name names a lexical variable or a local symbol macro.

 \Rightarrow add-variable-ignore client (environment environment) name ignore [Method]

This is the main method on add-variable-ignore. It calls describe-variable to obtain an existing variable description. It then calls merge-ignore to create a new variable description. Finally, it calls augment-with-variable-description in order to create and return a new environment.

Annotating a variable with a dynamic-extent declaration

 \Rightarrow add-variable-dynamic-extent client environment name [Generic Function]

This function returns a new environment that is like *environment* except that the variable named *name* has been annotated with an dynamic-extent declaration.

This function can be used only when *name* names a lexical variable.

 \Rightarrow add-variable-dynamic-extent client (environment environment) name [Method]

This is the main method on add-variable-dynamic-extent. It calls describe-variable to obtain an existing variable description. It then calls merge-dynamic-extent to create a new variable description. Finally, it calls augment-with-variable-description in order to create and return a new environment.

3.4.2 Adding and annotating functions

Adding a local function

 \Rightarrow add-local-function *client environment name* & optional *identity* [Generic Function]

This function returns a new environment that is like *environment* except that it has been augumented with a local function named *name*. The optional argument *identity* can be supplied by client code to distinguish different functions with the same name.

 \Rightarrow add-local-function *client* (*environment environment*) name & optional *identity* [Method]

This is the main method on add-local-function. It instantiates the class local-function-description and then creates a new environment by calling the function augment-with-function-description.

Adding a local macro

 \Rightarrow add-local-macro client environment name expander [Generic Function]

This function returns a new environment that is like *environment* except that it has been augmented with a local macro named **name**. The argument *expander* is a macro-expansion function that takes two arguments, a form and an environment.

 \Rightarrow add-local-macro client (environment environment) name expander [Method]

This is the main method on add-local-macro. It instantiates the class named local-macro-description and then creates a new environment by calling the function augment-with-function-description.

Annotating a function with a type

 \Rightarrow add-function-type client environment name type [Generic Function]

This function returns a new environment that is like *environment* except that the function named *name* has been annotated with the type specifier *type*.

The type of the function returned when the new environment is queried for the function named *name* will have a new type that is the conjunction of *type* and the type it had in *environment*.

This function can be used when name names a local function or a global function.

 \Rightarrow add-function-type client (environment environment) name type [Method]

This is the main method on add-function-type. It calls describe-function to obtain an existing function description. It then calls merge-type to create a new function description. Finally, it calls augment-with-function-description in order to create and return a new environment.

Annotating a function with an ignore declaration

 \Rightarrow add-function-ignore client environment name ignore [Generic Function]

This function returns a new environment that is like *environment* except that the function named *name* has been annotated with an **ignore** declaration.

The argument *ignore* must be the symbol **ignore** or the symbol **ignorable**.

This function can be used when *name* names a local function or a local macro.

 \Rightarrow add-function-ignore client (environment environment) name ignore [Method]

This is the main method on add-function-ignore. It calls describe-function to obtain an existing function description. It then calls merge-ignore to create a new function description. Finally, it calls augment-with-function-description in order to create and return a new environment.

Annotating a function with a dynamic-extent declaration

 \Rightarrow add-function-dynamic-extent client environment name [Generic Function]

This function returns a new environment that is like *environment* except that the function named *name* has been annotated with an dynamic-extent declaration.

This function can be used only when *name* names a local function.

30

 \Rightarrow add-function-dynamic-extent client (environment environment) name [Method]

This is the main method on add-function-dynamic-extent. It calls describe-function to obtain an existing variable description. It then calls merge-dynamic-extent to create a new variable description. Finally, it calls augment-with-function-description in order to create and return a new environment.

Annotating a function with an inline declaration

 \Rightarrow add-inline client environment name inline [Generic Function]

This function returns a new environment that is like *environment* except that the function named *name* has been annotated with an **inline** declaration.

The argument *inline* must be the symbol inline or the symbol notinline.

This function can be used when *name* names a local function or a local macro.

 \Rightarrow add-inline client (environment environment) name inline [Method]

This is the main method on add-inline. It calls describe-function to obtain an existing function description. It then calls merge-inline to create a new function description. Finally, it calls augment-with-function-description in order to create and return a new environment.

3.4.3 Adding blocks

 \Rightarrow add-block client environment name & optional identity [Generic Function]

This function returns a new environment that is like *environment* except that it has been augumented with a block named *name*, which must be a symbol. The optional argument *identity* can be supplied by client code to distinguish different blocks with the same name.

 \Rightarrow add-block client (environment environment) name & optional identity [Method]

This is the main method on add-block. It instantiates the class block-description and then creates a new environment by calling the function augment-with-block-description

Adding tags 3.4.4

 \Rightarrow add-tag client environment tag &optional identity [Generic Function]

This function returns a new environment that is like *environment* except that it has been augumented with a tag named tag, which must be a go tag, i.e. a symbol or an integer. The optional argument *identity* can be supplied by client code to distinguish different tags with the same name.

 \Rightarrow add-tag client (environment environment) tag &optional identity [Method]

This is the main method on add-tag. It instantiates the class tag-description and then creates a new environment by calling the function augment-with-tag-description.

Annotating the optimize qualities 3.4.5

Annotating optimize with a speed value

 \Rightarrow add-speed client environment value

This function returns a new environment that is like *environment* except that the optimize information has been updated with a speed quality value.

The argument *value* must be an integer between 0 and 3.

 \Rightarrow add-speed client (environment environment) value

This is the main method on add-speed. It calls describe-optimize to obtain the existing optimize description. It then calls merge-speed to create a new optimize description. Finally, it calls augment-with-optimize-description in order to create and return a new environment.

Annotating optimize with a compilation-speed value

 \Rightarrow add-compilation-speed *client environment value*

This function returns a new environment that is like *environment* except that the optimize information has been updated with a compilation-speed quality value.

[Generic Function]

[Generic Function]

[Method]

3.4. HIGH-LEVEL ANNOTATION FUNCTIONS

The argument *value* must be an integer between 0 and 3.

 \Rightarrow add-compilation-speed client (environment environment) value [Method]

This is the main method on add-compilation-speed. It calls describe-optimize to obtain the existing optimize description. It then calls merge-compilation-speed to create a new optimize description. Finally, it calls augment-with-optimize-description in order to create and return a new environment.

Annotating optimize with a debug value

 \Rightarrow add-debug client environment value

This function returns a new environment that is like *environment* except that the **optimize** information has been updated with a **debug** quality value.

The argument *value* must be an integer between 0 and 3.

 \Rightarrow add-debug client (environment environment) value [Method]

This is the main method on add-debug. It calls describe-optimize to obtain the existing optimize description. It then calls merge-debug to create a new optimize description. Finally, it calls augment-with-optimize-description in order to create and return a new environment.

Annotating optimize with a safety value

 \Rightarrow add-safety client environment value

This function returns a new environment that is like *environment* except that the **optimize** information has been updated with a **safety** quality value.

The argument *value* must be an integer between 0 and 3.

 \Rightarrow add-safety client (environment environment) value

This is the main method on add-safety. It calls describe-optimize to obtain the existing optimize description. It then calls merge-safety to create a new optimize description. Finally, it calls augment-with-optimize-description in order to create and return a new environment.

33

[Generic Function]

[Method]

[Generic Function]

[Generic Function]

Annotating optimize with a space value

This function returns a new environment that is like *environment* except that the **optimize** information has been updated with a **space** quality value.

The argument *value* must be an integer between 0 and 3.

 \Rightarrow add-space client (environment environment) value [Method]

This is the main method on add-space. It calls describe-optimize to obtain the existing optimize description. It then calls merge-space to create a new optimize description. Finally, it calls augment-with-optimize-description in order to create and return a new environment.

Bibliography

[Ste90] Guy L. Steele, Jr. Common LISP: The Language (2Nd Ed.). Digital Press, Newton, MA, USA, 1990.

Index

:class-name Initarg, 9 :compilation-speed Initarg, 11 :debug Initarg, 11 :dynamic-extent Initarg, 8 :expander Initarg, 9 :expansion Initarg, 8 :identity Initarg, 7 :ignore Initarg, 8 :inline Initarg, 10 :metaclass-name Initarg, 13 :method-class-name Initarg, 10 :name Initarg, 7 :safety Initarg, 12 :space Initarg, 12 :speed Initarg, 10 :superclass-names Initarg, 12 :type Initarg, 7 add-block Generic Function, 31 add-block Method, 31 add-compilation-speed Generic Func- add-safety Generic Function, 33 tion, 32add-compilation-speed Method, 33 add-debug Generic Function, 33 add-debug Method, 33 add-function-dynamic-extent Generic Func-tion, 26 tion, 30 add-function-dynamic-extent Methodadd-speed Generic Function, 32 31

add-function-ignore Generic Function, 30add-function-ignore Method, 30 add-function-type Generic Function, add-function-type Method, 30 add-inline Generic Function, 31 add-inline Method, 31 add-lexical-variable Generic Function. 26 add-lexical-variable Method, 26 add-local-function Generic Function, 29add-local-function Method, 29 add-local-macro Generic Function, 29 add-local-macro Method, 29 add-local-symbol-macro Generic Function, 27add-local-symbol-macro Method, 27 add-safety Method, 33 add-space Generic Function, 34 add-space Method, 34 add-special-variable Generic Funcadd-special-variable Method, 27 add-speed Method, 32

INDEX

add-tag Generic Function, 32 4 describe-optimize Generic Function, add-tag Method, 32 add-variable-dynamic-extent Generic Func-6 describe-tag Generic Function, 5 tion, 28add-variable-dynamic-extent Methoddescribe-variable Generic Function, 28add-variable-ignore Generic Funcdynamic-extent-mixin Class, 8 tion, 28 dynamic-extent Method, 8 add-variable-ignore Method, 28 expander-mixin Class, 9 add-variable-type Generic Function, expander Method, 9 27expansion-mixin Class, 8 add-variable-type Method, 27 expansion Method, 8 augment-with-block-description GenenicEumon-description Class, 14 tion, 20generic-function-description Class, augment-with-function-description 16Generic Function, 20 global-function-description Class, augment-with-tag-description Generic Func- 15 tion, 20global-macro-description Class, 16augment-with-variable-description global-symbol-macro-description Class, Generic Function, 19 15authentic-function-description Class, 7 identity Method, 7 authentic-variable-description $Class_{pgnore-mixin} Class, 8$ 13ignore Method, 8 block-description Class, 16 inline-mixin Class, 9 class-description Class, 17 inline Method, 10 class-name-mixin Class, 9 invalid-description-for-merging-compilationclass-name Method, 9 Condition, 24 compilation-speed-mixin Class, 11 invalid-description-for-merging-debug-information compilation-speed Method, 11 Condition, 24 $\texttt{constant-variable-description} Class \texttt{invalid-description-for-merging-dynamic-extended} and \texttt{constant-variable-description} and \texttt{constant-variable-descr$ 15Condition, 22 debug-mixin Class, 11 invalid-description-for-merging-ignore-inform debug Method, 11 Condition, 22 describe-block Generic Function, 5 invalid-description-for-merging-inline-inform describe-class Generic Function, 6 Condition, 23 describe-function Generic Function, invalid-description-for-merging-safety-inform

```
Condition, 25
                                     method-class-name-mixin Class, 10
invalid-description-for-merging-spacethidhfolranstinoame Method, 10
       Condition, 25
                                     name-mixin Class, 7
invalid-description-for-merging-speame Nifthrmation
       Condition, 23
                                     no-block-description Condition, 5
invalid-description-for-merging-type-dhasembesioniption Condition, 6
       Condition, 21
                                     no-function-description Condition,
lexical-variable-description Class,
                                             5
        14
                                     no-tag-description Condition, 5
local-function-description Class,
                                    no-variable-description Condition,
        15
local-macro-description Class, 16
                                     optimize-description Class, 17
local-symbol-macro-description Classafety-mixin Class, 12
        15
                                     safety Method, 12
macro-description Class, 14
                                     space-mixin Class, 11
merge-compilation-speed Generic Fungpace Method, 12
        tion, 24
                                     special-operator-description Class,
merge-compilation-speed Method, 24
                                             16
merge-debug Generic Function, 24
                                     special-variable-description Class,
merge-debug Method, 25
                                             14
merge-dynamic-extent Generic Func-
                                     speed-mixin Class, 10
        tion, 22
                                     speed Method, 10
\texttt{merge-dynamic-extent} \ Method, 22, 23 \ \texttt{superclass-names-mixin} \ Class, 12
merge-ignore Generic Function, 22
                                     superclass-names Method, 13
merge-ignore Method, 22
                                     symbol-macro-description Class, 13
merge-inline Generic Function, 23
                                     tag-description Class, 17
merge-inline Method, 23
                                     type-mixin Class, 7
merge-safety Generic Function, 25
                                     type Method, 7
merge-safety Method, 26
                                     variable-description Class, 13
merge-space Generic Function, 25
                                     augment-with-block-description
merge-space Method, 25
                                             Method, 20
merge-speed Generic Function, 23
                                     augment-with-function-description
merge-speed Method, 23
                                             Method, 20
merge-type Generic Function, 21
                                     augment-with-optimize-description
merge-type Method, 21
                                             Generic Function, 21
metaclass-name-mixin Class, 13
                                     augment-with-optimize-description
metaclass-name Method, 13
                                             Method, 21
```

INDEX

```
augment-with-tag-description
       Method, 20
augment-with-variable-description
       Method, 19
merge-compilation-speed
       Method, 24
merge-debug
       Method, 25
merge-ignore
       Method, 22
merge-inline
       Method, 23
merge-safety
       Method, 26
merge-space
       Method, 25
merge-speed
       Method, 24
merge-type
       Method, 22
```