CHAPTER 1

Getting Started

This appendix gives instructions on how to install and use the Marea system.

1.1 Installation

There are two possibilities to install Marea. One is downloading an already prepared environment and the other is to install Marea in a standard Pharo distribution.

1.1.1 Downloading a one-click image

1. Download the one-click Marea distribution from https://gforge.inria.fr/frs/download.php/31256/Pharo-1.4-Marea.zip.

2. Launch the executable of your platform:

   - Mac: Marea.app
   - Linux: Marea.app/Marea.sh
   - Windows: Marea.app/Marea.exe

1.1.2 Building a custom image

To install Marea in a Pharo environment:

1. Get a Pharo image from http://www.pharo-project.org/. The tested versions of Pharo with Marea are 1.4 and 2.0.

2. Open a Workspace and evaluate the following code:

```
Gofer it
url: 'http://ss3.gemstone.com/ss/Marea';
package: 'ConfigurationOfMarea';
load.

(Smalltalk at: #ConfigurationOfMarea) project stableVersion load.
```

   This will install Marea and all its dependencies such as Ghost and Fuel.

1.2 Examples

In this section we present some examples to show how to use Marea.

1.2.1 Swapping Regular Objects

The API of Marea is very straightforward and minimalistic. For example, the following line swaps out a regular graph:

```
| today yesterday |
today := Date today.
MARObjectGraphExporter new swapOutGraph: today.
yesterday := Date yesterday.
MARObjectGraphExporter new swapOutGraph: yesterday.
```

The default persistency strategy of Marea stores each swapped graph in a separate file located in a directory called *swapSpace*. Such directory is created in the directory where the Pharo image is. For example, if the Pharo image is in */Users/mariano/marea/marea.image* then the directory is */Users/mariano/marea/swapSpace*.

The name of the file created for each graph is its graph ID and the file extension is *.swap*. In our example, today is written into a file named *1.swap*. The next swapped graph yesterday in *2.swap* and so on.

Since the swapping in is automatic, a developer does not have to use it explicitly. As soon as one of the proxies of the graph is used, the graph is swapped in. In our example, if we do:

```
today asSeconds.
```

The proxy intercepts the message asSeconds and the graph is swapped in and answers the value of today as seconds.

The object passed to swapOutGraph: can be any type of object. So for example we can also do:

```
MARObjectGraphExporter new swapOutGraph: Date allInstances.
```

In this case we swap out all instances of Date. We can also do:

```
MARObjectGraphExporter new swapOutGraph: (
    Array
        with: Date allInstances
        with: DateAndTime allInstances
        with: (TextStyle named: 'Bitmap DejaVu Sans')
        with: UITheme currentSettings ).
```

Here we swap out all instances of Date and DateAndTime, the fonts named 'Bitmap DejaVu Sans' and the current settings of the UI theme.

We could continue giving more examples, but we hope it is clear that the method swapOutGraph: provides the swapping mechanism for almost any type of object graph. In the next section we show how to swap code.

1.2.2 Swapping Classes

Marea also allows the developer to swap classes. To accomplish that, Marea's API provides some helper methods such as swapOutClass:, swapOutClassAndSubclasses:, swapOutClassWithInstances: and swapOutClassWithSubclassesWithInstances:. Examples:

```
"Here we just swap the class DateAndTime"
MARObjectGraphExporter new swapOutClass: DateAndTime.
"Here we swap the class DateAndTime and all its instances"
MARObjectGraphExporter new swapOutClassWithInstances: DateAndTime.
"Here we swap the class DateAndTime and its subclasses (TimeStamp)"
MARObjectGraphExporter new swapOutClassAndSubclasses: DateAndTime.
"Here we swap the class DateAndTime, TimeStamp and all their instances"
MARObjectGraphExporter new swapOutClassWithSubclassesWithInstances: DateAndTime.
```

We can also, for instance, swap out some development tools:

```
classesToSwap := { TestRunnerBrowser. SystemBrowser. FlatMessageListBrowser. HierarchyBrowser.
    ClassListBrowser. ChangedMessageSet. MessageNames. ProtocolBrowser. RecentMessageSet. ChangeList.
    VersionsBrowser. ChangeSorter. ChangeSetBrowser. DualChangeSorter. TimeProfiler. TimeProfileBrowser.
    PointerExplorer. ViewHierarchyExplorer. ObjectExplorerWrapper. ViewHierarchyExplorer.
    FileContentsBrowser. FileList. Inspector. ProcessBrowser. Debugger. PreDebugWindow. TimeProfileBrowser.
    TimeProfiler. Finder. FinderUI. FinderNode. }.

classesToSwap do: [:each | MARObjectGraphExporter new swapOutClassWithSubclassesWithInstances: each].
```

Or we can swap all UI themes:

```
classesToSwap := { BlueUITheme. UIThemeStandardSqueak. UIThemeVistary. UIThemeW2K.
    GLMOrangeUITheme. VistaryThemeIcons. WateryThemeIcons. }.
classesToSwap do: [:each | MARObjectGraphExporter new swapOutClassWithSubclassesWithInstances: each].
```

There is also a helper method to swap out traits (groups of methods that act as a unit of reuse from which classes are composed):

```
MARObjectGraphExporter new swapOutTrait: TAssertable.
```

Finally, we can also swap a whole package, which means swapping all its classes:

```
MARObjectGraphExporter new swapOutPackage: (PackageInfo named: 'AST').
```

1.2.3 Using Different Storage Engines

By default, Marea has a persistency strategy based on files in the local file system. To install persistency strategies based on No-SQL databases one has explicitly tell the installer to do so:

```
(Smalltalk at: #ConfigurationOfMarea) project stableVersion load: 'MareaRiak'.
```

or:

```
(Smalltalk at: #ConfigurationOfMarea) project stableVersion load: 'MareaMongoDB'.
```

That installs the Marea backend for those databases and also the database driver for them. Once we have install them we can set them as the storage engine strategy by doing:

```
MARObjectGraphSwapper graphPersistencyStrategy: MARMongoDBGraphPersistency.
```

or:

```
MARObjectGraphSwapper graphPersistencyStrategy: MARRiakDBGraphPersistency.
```

We can also change the configuration of those databases. For example, for Riak, the default connection is on *http://localhost:8098/riak*. In MongoDB it is *(Mongo host: 'localhost' port: 27017) open*. To change it:

```
MARRiakDBGraphPersistency databaseConnection: 'http://somedomain:PPPP/riak'.
MARMongoDBGraphPersistency databaseConnection: (Mongo host: 'somedomain' port: PPPP) open.
```

1.2.4 Rest of the API

Resetting. Marea provides a method to reset all its internal configuration and data. This leaves the environment as clean as it was before starting to do anything with Marea. It is useful for testing. The method MARObjectGraphSwapper resetAll resets all the internal tables mentioned in Chapter **??** such as the GraphTable and SharedProxiesTable.

```
MARObjectGraphSwapper resetAll.
```

It also resets the counter to get the graph ID, swaps in all swapped out graphs and removes all stored graphs remaining in the storage engine. The methods used internally by resetAll can also be used directly by the user. Examples:

```
MARObjectGraphSwapper uninstallAllProxies.
MARObjectGraphSwapper resetGlobalTable.
MARObjectGraphSwapper resetSharedProxiesTable.
MARObjectGraphSwapper resetGraphID.
MARObjectGraphSwapper resetGraphStorage.
```

Cleaning. Marea could clean the system after every swap in. However, to avoid adding extra overhead, we let the user decide when to clean Marea. By cleaning Marea we mean to compact the tables GraphTable and SharedProxiesTable to release even more memory.

```
MARObjectGraphSwapper clean.
```

Logging. The user of Marea can decide whether to log or not and also *where* to log. If log is enable, Marea logs information when a graph is swapped out and when it is swapped in. In the latter it also logs the sequence of messages (stack trace) that triggered the swap in. Hence, the logging adds a significant overhead to Marea's performance. Because of that, we disable the logging by default for general-purpose usage and the user has to enable it when needed:

```
MARLogger createAndSetLogger.
```

That method creates, by default, a file *MareaLog.txt* which is placed in the image directory. In addition, the user can choose to log into the Pharo Transcript instead of files or to directly disable again the log:

```
MARLogger changeToFileLogger.
MARLogger changeToTranscriptLogger.
MARLogger disableLogging.
```