Config: Installation and Search Paths

Version 5.1

February 14, 2011

1 Configuring Directories and Search Paths

```
(require config)
```

The `config` library specifies the location of directories (such as the main documentation directory) and also directory search paths (such as a list of directories to search for documentation).

Note: Instead of `require`ing `config` directly, use the `setup/dirs` library, which combines information from `config` and other sources.

The `config` module must export the following values. In all cases where a `delay`ed value is expected for an exported identifier, the value can be a `delay`ed `#f` to indicate the default.

`doc-dir` : `(promise/c (or/c path? string? bytes? false/c))`

A `delay`ed path, string, or byte string for the main documentation directory. It defaults to a `"doc"` sibling directory of the main collection directory.

`lib-dir` : `(promise/c (or/c path? string? bytes? false/c))`

A `delay`ed path, string, or byte string for the main directory containing C libraries and build information; it defaults to a `"lib"` sibling directory of the main collection directory.

`dll-dir` : `(promise/c (or/c path? string? bytes? false/c))`

A `delay`ed path, string, or byte string for a directory containing Unix shared libraries for the main executable; it defaults to the main C-library directory

---

`include-dir` : `(promise/c (or/c path? string? bytes? false/c))`

A `delay`ed path, string, or byte string for the main directory containing C header files; it defaults to an `"include"` sibling directory of the main collection directory.

`bin-dir` : `(promise/c (or/c path? string? bytes? false/c))`

A `delay`ed path, string, or byte string for the main directory containing executables; it defaults to a `"bin"` sibling directory of the main collection directory.

`doc-search-dirs` : `(promise/c (or/c path? string? bytes? false/c))`

A `delay`ed path, string, byte string, or `#f` representing the search path for documentation;

each `#f` in the list, if any, is replaced with the default search path, which is the user- and version-specific `"doc"` directory followed by the main documentation directory.

`lib-search-dirs` : `(promise/c (or/c path? string? bytes? false/c))`

Like `doc-search-dirs`, but for directories containing C libraries and other build information

---

`include-search-dirs` : `(promise/c (or/c path? string? bytes? false/c))`

Like `doc-search-dirs`, but for directories containing C header files

`absolute-installation?` : `boolean?`

A (simple, non-`delay`ed) boolean that is `#t` if the installation uses absolute path names, `#f` otherwise.

`cgc-suffix` : `(promise/c (or/c string? false/c))`

A `delay`ed string used as the suffix (before the actual suffix, such as `".exe"`) for a `"CGC"` executable. Use Windows-style casing, and the string will be downcased as appropriate (e.g., for a Unix binary name). A `#f` value means that if the `mzscheme` binary identifies itself as CGC, then the suffix is `""`, otherwise it is `"CGC"`.

`3m-suffix` : `(promise/c (or/c string? false/c))`

Analogous to `cgc-suffix`, but for 3m. A `#f` value means that if the `"mzscheme"` binary identifies itself as CGC, then the suffix is `"3m"`, otherwise it is `""`.

2   Overriding the Installation's Configuration

A user can override an installation's configuration through a `"config"` collection in the user's collection directory (which normally takes precedence over the main collection directory).

3 Configuration Language

```
(require setup/configtab)
```

The `setup/configtab` library defines a language module that can be used to implement `config`.

When `setup/configtab` is used as a language module, the module body must consist of a sequence of

```
(define id val)
```

declarations, where each `id` is one of the names that the `config` library must export, and `val` is an expression for the value (which will be automatically wrapped with `delay` when needed). If a required export has no corresponding `define`, a definition with `#f` is inserted automatically.