template.pl

Tim Menzies
Lane Deptartment of Computer Science and Electrical Engineering
West Virginia University
Morgantown, WV 26506
`tim@menzies.us`

Contents

1 Template

Standard Prolog file structure */

/*

2 Header */

/*

2.1 Loads

Load some code using ”ensure_loaded”. e.g.: */

```
:- ensure_loaded([xx]).
```

/*

(Warning: there is some quirk in the Prolog loader when doing `goal_expansion`s. If you are using `ecg`s or `defs`, then DON’T `ensure_loaded` those files; rather, load them the old-fashioned way: */

```
:- [ecg]. */
```

2.2 Operators

Change the Prolog parser using `op/3`x. e.g.: */

```
:- op(700,xfx,and).
```

/*

2.3 Flags

Index predicates (with `index/1`) or make some dynamic (with `dynamic/1`) or discontiguous (with `discontiguous/1`) or .... e.g.: */

```
:- index(myemp(1,1,0,1,00)).
:- discontiguous defs/2.
:- mulitfile defs/2.
:- dynamic defs/2.
```

/*

2.4 Hooks

Change the Prolog parser using `term_expansion/2` and `goal_expansion/2`. e.g. */

```
term_expansion(X and Y,Z) :- xpands(X and Y, Z).

goal_expansion(L with M and N,Z) :- tupleCompile(L with M and N,Z).
```

/*

2.5 Hacks

We won’t talk about these. I didn’t not write these. You did not see them. */

```
goal_expansion(true(X,X),true).
```

/*

3 Body

Insert your applciation code here.

3.1 Utils

Finally, add any general utilities last. e.g. */

```
barph(X) :- format('W> ~w\n',X), fail. /*
```

4 Footer */

/*

4.1 Start-ups

After all the above is loaded, how do we get this stuff started. */