defensive-to-contracts

Version 7.8

Peter Zhong

October 14, 2020

```
(require defensive-to-contracts)
      package: defensive-to-contracts
```

This tool is highly experimental and has not been tested on a lot of codebases.

This package presents a tool that converts defensive programming into equivalent contracts. Currently, the tool is still at its infancy and have not been tested on many code bases. The sample folder contains some example files to get you started. So far I have exposed two functions that allows you to input a path and either returns the raw result or load up a GUI that allows you to change a file.

```
(contract-infos-on-path path) → (listof func-contract-info?)
  path : path-string?
```

Process the file located in *path* and returns the result as a list of func-contract-info where each element represents a procedure in the file in the order that the procedures appeared.

```
(path-addcontracts-withGUI path) → void?
  path : path-string?
```

Process the file located in *path* and launch a GUI that allows for the contracts to be viewed and applied.

```
(func-contract-info? v) → boolean?
  v : any/c
```

Returns #t if *v* is a func-contract-info.

```
(func-contract-info func-name
                    path
                    spanset
                    contract
                    define-end
                    body-start
                    desirability) → func-contract-info?
  func-name : any/c
  path : string?
  spanset : character-set?
  contract : any/c
  define-end : integer?
  body-start : integer?
  desirability : integer?
```

This procedure initialises a struct `func-contract-info`.

```
(serializable-struct func-contract-info (func-
name path spanset contract define-end body-start desirability)
#:guard (struct-guard/c any/c path-string? character-
set? any/c integer? integer? integer?))
```

func-name refers to the name of the function. *path* path refers to the path of the file. *spanset* is a `character-set` denoting which character to delete. *contract* contains an s expression of the contract generated. *define-end* and *body-start* refers to the position of the end of the define in a procedure and the start of the body. These two location essentially points to where the contracts are added. *desirability* is an internal measure of how desirable the contract generated is.

```
(character-set? v) → boolean?
  v : any/c
```

Returns `#t` if *v* is a `character-set`.

```
(character-set s) → character-set?
  s : set?
```

Returns a `character-set` from a `set` of integers.

```
(charset-size s) → integer?
  s : character-set?
```

```
(charset-empty? s) → boolean?
  s : character-set?
```

```
(empty-charset) → character-set?
```

```
(union-charset c1 c2) → character-set?
  c1 : character-set?
  c2 : character-set?
```

```
(subtract-charset c1 c2) → character-set?
  c1 : character-set?
  c2 : character-set?
```

This function returns a character set with characters included in *c1* but not in *c2*