NAME

pkg_repos, **pkg_repos_total_count**, **pkg_repos_activated_count**, **pkg_repo_url**, **pkg_repo_ident**, **pkg_repo_name**, **pkg_repo_ident_from_name**, **pkg_repo_key**, **pkg_repo_fingerprints**, **pkg_repo_signature_type**, **pkg_repo_mirror_type**, **pkg_repo_enabled**, **pkg_repo_priority**, **pkg_repo_find_ident**, **pkg_repo_find_name** - manipulate repositories

LIBRARY

library "libpkg"

SYNOPSIS

#include <pkg.h>

int
pkg_repos(*struct pkg_repo **r*);

int
pkg_repos_total_count(*void*);

int
pkg_repos_activated_count(*void*);

*const char **
pkg_repos_url(*struct pkg_repo **);

*const char **
pkg_repo_ident(*struct pkg_repo **);

*const char **
pkg_repo_name(*struct pkg_repo **);

*const char **
pkg_repo_ident_from_name(*const char **);

*const char **
pkg_repo_key(*struct pkg_repo **);

*const char **
pkg_repo_fingerprints(*struct pkg_repo **);

signature_t

pkg_repo_signature_type(*struct pkg_repo *);

bool
pkg_repo_enabled(*struct pkg_repo *);

unsigned int
pkg_repo_priority(*struct pkg_repo *);

mirror_t
pkg_repo_mirror_type(*struct pkg_repo *);

*struct pkg_repo **
pkg_repo_find_ident(*const char *ident*);
;
*struct pkg_repo **
pkg_repo_find_name(*const char *name*);
;

DESCRIPTION

pkg_repos() Takes the address of a pointer to the repository. The pointer should be initialised to NULL before being passed to the function, on each iteration the r will point to the new repository. Returns EPKG_FATAL if an error occurred, otherwise return EPKG_OK until the last repository is found in that case EPKG_END is returned.

pkg_repos_total_count() returns the total number of defined repositories.

pkg_repos_activated_count() returns the total number of "enabled" repositories.

pkg_repo_url() takes a pointer to a repository as argument and returns the "url" defined for this repository.

pkg_repo_ident() takes a pointer to a repository as argument and returns the internal identification string of the repository.

pkg_repo_name() takes a pointer to a repository as argument and returns the "name" of the repository.

pkg_repo_key() takes a pointer to a repository as argument and returns the path to the public key. If the repository is not signed by a public key, NULL will be returned.

pkg_repo_fingerprints() takes a pointer to a repository as argument and returns the path to the

fingerprints. If the repository is not signed using the "FINGERPRINT" method, NULL will be returned.

pkg_repo_signature_type() Take a pointer to a repository as argument and return the type of signature it uses.

SIG_NONE The repository is not signed

SIG_PUBKEY The repository is signed using the PUBKEY method

SIG_FINGERPRINT The repository is signed using the FINGERPRINT method

pkg_repo_enabled() takes a pointer to a repository as argument and returns **true** if the repository is "enabled". Otherwise, return **false**.

pkg_repo_priority() takes a pointer to a repository as argument and returns the priority of the repository as an unsigned integer. Packages ae chosed preferentially from the repository with the highest priority value and which has that package available.

pkg_repo_mirror_type() takes a pointer to a repository as argument and returns the type of mirroring it uses.

SRV The repository is using SRV record query to get the mirrors.

HTTP The repository is using the HTTP query method to get the mirrors.

NOMIRROR The repository does not have any mirror.

pkg_repo_find_ident() Take an internal indentification string as argument and return a pointer to a repository. If no repository matches, NULL is returned.

pkg_repo_find_name() Take a repository "name" as argument and return a pointer to a repository. If no repository matches, NULL is returned.