#include <stdlib.h>

#include <stdio.h>

#include <string.h>

#include "option\_list.h"

#include "utils.h"

#include "data.h"

list \*read\_data\_cfg(char \*filename)

{

FILE \*file = fopen(filename, "r");

if(file == 0) file\_error(filename);

char \*line;

int nu = 0;

list \*options = make\_list();

while((line=fgetl(file)) != 0){

++nu;

strip(line);

switch(line[0]){

case '\0':

case '#':

case ';':

free(line);

break;

default:

if(!read\_option(line, options)){

fprintf(stderr, "Config file error line %d, could parse: %s\n", nu, line);

free(line);

}

break;

}

}

fclose(file);

return options;

}

metadata get\_metadata(char \*file)

{

metadata m = { 0 };

list \*options = read\_data\_cfg(file);

char \*name\_list = option\_find\_str(options, "names", 0);

if (!name\_list) name\_list = option\_find\_str(options, "labels", 0);

if (!name\_list) {

fprintf(stderr, "No names or labels found\n");

}

else {

m.names = get\_labels(name\_list);

}

m.classes = option\_find\_int(options, "classes", 2);

free\_list(options);

if(name\_list) {

printf("Loaded - names\_list: %s, classes = %d \n", name\_list, m.classes);

}

return m;

}

int read\_option(char \*s, list \*options)

{

size\_t i;

size\_t len = strlen(s);

char \*val = 0;

for(i = 0; i < len; ++i){

if(s[i] == '='){

s[i] = '\0';

val = s+i+1;

break;

}

}

if(i == len-1) return 0;

char \*key = s;

option\_insert(options, key, val);

return 1;

}

void option\_insert(list \*l, char \*key, char \*val)

{

kvp\* p = (kvp\*)xmalloc(sizeof(kvp));

p->key = key;

p->val = val;

p->used = 0;

list\_insert(l, p);

}

void option\_unused(list \*l)

{

node \*n = l->front;

while(n){

kvp \*p = (kvp \*)n->val;

if(!p->used){

fprintf(stderr, "Unused field: '%s = %s'\n", p->key, p->val);

}

n = n->next;

}

}

char \*option\_find(list \*l, char \*key)

{

node \*n = l->front;

while(n){

kvp \*p = (kvp \*)n->val;

if(strcmp(p->key, key) == 0){

p->used = 1;

return p->val;

}

n = n->next;

}

return 0;

}

char \*option\_find\_str(list \*l, char \*key, char \*def)

{

char \*v = option\_find(l, key);

if(v) return v;

if(def) fprintf(stderr, "%s: Using default '%s'\n", key, def);

return def;

}

char \*option\_find\_str\_quiet(list \*l, char \*key, char \*def)

{

char \*v = option\_find(l, key);

if (v) return v;

return def;

}

int option\_find\_int(list \*l, char \*key, int def)

{

char \*v = option\_find(l, key);

if(v) return atoi(v);

fprintf(stderr, "%s: Using default '%d'\n", key, def);

return def;

}

int option\_find\_int\_quiet(list \*l, char \*key, int def)

{

char \*v = option\_find(l, key);

if(v) return atoi(v);

return def;

}

float option\_find\_float\_quiet(list \*l, char \*key, float def)

{

char \*v = option\_find(l, key);

if(v) return atof(v);

return def;

}

float option\_find\_float(list \*l, char \*key, float def)

{

char \*v = option\_find(l, key);

if(v) return atof(v);

fprintf(stderr, "%s: Using default '%lf'\n", key, def);

return def;

}